Environnement Canada Environment Canada

Institut national de recherche en hydrologie

RAPPORT N° 39 DE L'INRH

ÉTUDE N° 155, COLLECTION DES RAPPORTS TECHNIQUES DE LA DGEI

Simulateur numérique de l'écoulement et du transfert de masse dans des réseaux stochastiques de fractures

A. Rouleau*

*Maintenant au Département des Sciences Appliquées
Université du Québec à Chicoutimi
555 boulevard de l'Université
Chicoutimi (Québec) G7H 2B1

INRH

**DIRECTION GÉNÉRALE DES EAUX INTÉRIEURES
INSTITUT NATIONAL DE RECHERCHE EN HYDROLOGIE
CENTRE NATIONAL DE RECHERCHE EN HYDROLOGIE
SASKATOON (SASKATCHEWAN) 1988**

(Available in English on request)

Publié avec l'autorisation
du ministre de l'Environnement

© Ministre des Approvisionnements et Services Canada 1988

Nº de cat. En 36-503/155F

ISBN 0-662-94974-9

Table des matières

		Page
RESUME		vii
ABSTRACT		vii
REMERCIEMENTS		viii
1.	INTRODUCTION	1
2.	SIMULATION STOCHASTIQUE DE SYSTEMES DE FRACTURES: PROGRAMMES NETWRK ET APEGEN	4
2.1	Introduction	4
2.2	Programme NETWRK : routine principale et données d'entrée	5
2.3	Génération du réseau de lignes	7
2.4	Génération de données d'espacement	10
2.5	Intersections de lignes	10
2.6	Génération d'un fichier de données pour le traçage de courbes	11
2.7	Détermination des éléments de lignes	12
2.8	Exemple simple	14
2.9	Génération de distributions d'ouvertures supplémentaires : Programme APEGEN	16
2.10	Conclusion	16
3.	SIMULATION NUMERIQUE DE L'ECOULEMENT DE FLUIDE DANS UN RESEAU BIDIMENTIONNEL DE FRACTURES DISCRETES : PROGRAMME NETFLO	18
3.1	Introduction	18
3.2	Formulation du modèle	20
3.3	Programme NETFLO	22
	3.3.1 Opérations préliminaires	22
	3.3.2 Calculs des paramètres de l'écoulement du fluide	28
3.4	Vérification du programme et évaluation de la précision	31
3.5	Conclusion	36
4.	SIMULATION STOCHASTIQUE DE LA MIGRATION DES PARTICULES DANS UN RESEAU DE FRACTURES : PROGRAMME NETRANS	37
4.1	Introduction	37
4.2	Programme NETRANS	38
	4.2.1 Détermination de la direction de l'écoulement	39
	4.2.2 Détermination de la vitesse d'écoulement moyenne dans chaque direction	42
	4.2.3 Nombre de particules requises	44
4.3	Conclusion	44
5.	CONCLUSION GENERALE	45
REFERENCES		46

Table des matières (suite)

Page

Annexe A : LISTE DES PRINCIPALES VARIABLES UTILISEES DANS LES PROGRAMMES NETWRK (NW), APEGEN (AP), NETFLO (NF) ET NETRANS (NT)..... 51
A.1 Liste des paramètres d'entrée..... 53
A.2 Liste des paramètres d'utilisation interne..... 57

Annexe B : FICHIERS D'ENTREE/SORTIE..... 67

Annexe C : STRUCTURE DES DONNEES D'ENTREE..... 73
C.1 Fichier d'entrée général (unité 1)..... 75
C.2 Fichier d'entrée supplémentaire pour le programme APEGEN (unité 21)..... 76

Annexe D : CONVENTIONS POUR LES ANGLES, LES DIRECTIONS, ET LA CONFIGURATION DES LIMITES..... 79
D.1 Conventions pour les angles..... 81
D.2 Conventions pour les cosinus directeurs calculés dans le programme APEGEN..... 81
D.3 Conventions et restrictions pour la configuration des limites..... 83

Annexe E : DIMENSIONNEMENT DES TABLEAUX..... 85

Annexe F : EXEMPLES D'EXECUTION..... 93
F.1 Modèle rectangulaire dont les fractures ne se terminent pas sur d'autres fractures: Réalisation 990..... 95
F.2 Modèle rectangulaire avec fractures se terminant sur d'autres fractures : Réalisation 992..... 116
F.3 Modèle circulaire : Réalisation 991..... 118

Annexe G : LISTES DES PROGRAMMES..... 121
G.1 Liste du programme NETWRK..... 123
G.2 Liste du programme APEGEN..... 166
G.3 Liste du programme NETFLO..... 171
G.4 Liste du programme NETRANS..... 198

Tableaux

2.1 Résultats choisis pour le réseau de lignes de la Figure 2.7..... 16

B.1 Liste de tous les fichiers d'entrée/sortie..... 69

B.2 Fichiers d'entrée et options de fichiers de sortie pour chaque programme..... 71

Illustrations

Page

Figure 2.1. Structure du programme NETWRK..... 5

Figure 2.2. Organigramme de la routine principale du programme NETWRK..... 6

Figure 2.3. Configuration du modèle : (A) modèle rectangulaire, (B) modèle circulaire..... 7

Figure 2.4. Organigramme du sous-programme GENLIN..... 8

Figure 2.5. Organigramme du sous-programme INTERS..... 12

Figure 2.6 Organigramme du sous-programme ELTDEF..... 13

Figure 2.7. Réseau de lignes simples généré par le programme NETWRK..... 15

Figure 2.8. Organigramme de la routine principale du programme APEGEN..... 17

Figure 3.1. Réseau de fractures composé de cinq segments connectés..... 21

Figure 3.2. Structure du programme NETFLO..... 23

Figure 3.3. Organigramme de la routine principale du programme NETFLO..... 24

Figure 3.4. Organigramme du sous-programme OPSTOR..... 25

Figure 3.5. Matrice symétrique à largeur de bande variable..... 27

Figure 3.6. Organigramme du sous-programme SOLPHI..... 29

Figure 3.7. Organigramme du sous-programme FLOCAL..... 31

Figure 3.8. Système de fractures utilisé pour évaluer la précision du programme NETFLO..... 32

Figure 3.9. Distribution de l'erreur d'arrondi cumulative pour différents nombres de noeuds entre les limites à charge fixe du système de la figure 3.8; (A) erreur absolue sur la charge hydraulique, (B) erreur relative sur la charge hydraulique, (C) erreur sur le débit..... 34

Figure 3.10. Erreur relative sur l'équilibre des masses en fonction du nombre de noeuds entre les limites à charge fixe du système de la figure 3.8..... 35

Illustrations (suite)

Page

Figure 4.1. Structure du programme NETRANS..... 38

Figure 4.2. Organigramme de la routine principale du programme NETRANS..... 39

Figure 4.3. Création du tableau IDIR (100) en vue de la détermination stochastique de la direction d'écoulement..... 40

Figure 4.4. Organigramme du sous-programme TTIMER..... 41

Figure 4.5. Système de fractures utilisé pour la vérification du programme NETRANS; toutes les fractures ont une ouverture de 5 µm..... 42

Figure 4.6. Comparaison de l'écoulement sous forme de piston déterministe et du temps de parcours stochastique de particules avec différents facteurs de pondération pour la vitesse d'écoulement..... 43

Figure D.1. Conventions pour les angles..... 81

Figure D.2. Conventions pour les cosinus directeurs calculés dans le programme APEGEN..... 82

Figure D.3. Numérotage des limites d'écoulement : (A) modèle rectangulaire, (B) modèle circulaire..... 83

Figure F.1. Conditions aux limites pour la réalisation 990..... 95

Figure F.2. Graphique tracé par ordinateur pour la réalisation 990; les numéros en petits caractères désignent les noeuds, les numéros en caractères plus gros désignent les fractures mais sans indiquer la famille; ces numéros ne sont pas imprimés lorsque NUMNPT > 200 ou IPRT =1..... 97

Figure F.3. Diagramme circulaire du fichier SUMFLO complété à la main..... 111

Figure F.4. Graphique tracé par ordinateur pour la réalisation 992; voir figure F.2 pour la signification des numéros..... 117

Figure F.5. Conditions aux limites pour la réalisation 991..... 118

Figure F.6. Graphique tracé par ordinateur pour la réalisation 991; voir figure F.2 pour la signification des numéros..... 120

Résumé

Le rapport décrit un progiciel de modélisation de fractures discrètes stochastiques (FDS) qui simule l'écoulement de l'eau souterraine et le transfert des contaminants dans des réseaux de fractures. Le programme NETWRK génère des réseaux de fractures bidimensionnels à l'aide d'une méthode de Monte Carlo basée sur des valeurs recueillies sur le terrain des paramètres statistiques de configuration des fractures, c'est-à-dire l'orientation, la longueur des traces, l'ouverture et la densité des fractures. Le programme APEGEN permet de générer des distributions d'ouvertures supplémentaires pour le réseau de fractures généré par le programme NETWRK. Un troisième programme, NETFLO, calcule l'écoulement fluide en régime permanent dans le réseau de fractures généré par NETWRK et APEGEN. Il calcule aussi certains paramètres statistiques dans chaque intervalle de direction de dix degrés, notamment la longueur totale de segments de fractures, la vitesse d'écoulement totale et le débit total. Ces derniers paramètres directionnels sont utilisés par le programme NETRANS pour calculer le temps de transit des particules sur une distance arbitraire à l'aide d'une méthode stochastique de deuxième niveau.

Abstract

This report describes a stochastic discrete fracture (SDF) modelling package that simulates ground-water flow and mass transport in fracture systems. The program NETWRK generates two-dimensional fracture networks with a Monte Carlo method based on the statistics of field data on fracture geometry, i.e., fracture orientation, trace length, aperture, and density. The program APEGEN can be used to generate supplementary aperture distributions for the fracture network generated by NETWRK. A third program, NETFLO, computes the steady-state fluid flow through the fracture network generated by NETWRK and APEGEN. NETFLO also computes the statistics of selected parameters in every ten-degree range of direction, including the total length of fracture segments, the total flow velocity, and the total flow rate. The latter directional parameters are used by the program NETRANS to compute the transit time of particles over an arbitrary distance using a second-level stochastic process.

Remerciements

Certaines parties des programmes informatiques dont il est question dans le présent document ont été écrites dans le cadre de ma thèse de Ph.D. à l'Université de Waterloo. Les nombreux conseils techniques et scientifiques que m'a donnés mon directeur de thèse, John E. Gale, maintenant à l'Université Memorial de Terre-Neuve, m'ont été grandement profitables. De nombreuses discussions avec Kent S. Novakowski et Ken G. Raven sur les processus de transfert dans les roches fracturées ont été une motivation pour l'écriture du programme NETRANS. D. Lentz a écrit une grande partie du programme NETRANS, et R. MacLeod a participé à l'écriture du programme APEGEN. L'aide financière pour l'écriture des programmes NETWRK et NETFLO provenait de trois sources principales : 1) une subvention de recherche accordée par le Conseil de recherches en sciences naturelles et en génie du Canada à J.E. Gale, lorsqu'il était à l'Université de Waterloo; 2) des bourses de la F.C.A.C. du ministère de l'Education du Québec; 3) une bourse de la North American Life Insurance Company accordée par l'intermédiaire du Conseil canadien des ingénieurs professionnels. Des modifications au programme NETWRK original et l'écriture du programme APEGEN ont été réalisées avec l'aide financière de la société Fracflow Consultants Inc. de St. John's (Terre-Neuve). La Direction générale des eaux intérieures a fourni une aide financière pour l'écriture du programme NETRANS et la préparation de la documentation de l'ensemble du progiciel de simulation de fractures discrètres stochastiques (FDS).

Introduction

Au cours de la dernière décennie, dans les pays industrialisés, les centres d'intérêt en hydrogéologie sont passés des questions relatives à la quantité d'eau souterraine aux problèmes relatifs à la qualité de l'eau souterraine et au transport des polluants. Par exemple, dans le domaine de l'écoulement de l'eau souterraine dans les excavations, les études hydrogéologiques portaient essentiellement, par le passé, sur le volume total d'eau qui s'écoulait dans une excavation (p. ex., Fawcet et coll., 1984). De même, dans l'évaluation des ressources en eau souterraine et dans l'exploitation des réservoirs d'hydrocarbures, en général on s'intéressait surtout au rendement des formations géologiques. Par conséquent, dans ces problèmes classiques de géologie appliquée, les paramètres hydrauliques d'intérêt étaient principalement la perméabilité globale et le coefficient d'emmagasinement, étant donné que ces paramètres déterminent le débit et le volume de fluide disponible pour un gradient hydraulique donné. Les modèles hydrogéologiques élaborés pour l'étude de ces problèmes dans les roches fracturées sont notamment des modèles basés sur le principe du continuum (Snow, 1969; Duguid et Lee, 1977), et des modèles de fractures discrètes (Wittke, 1970). On a analysé la contribution à la perméabilité d'une masse rocheuse des réseaux réguliers de fractures de taille finie à l'aide de modèles de fractures discrètes (Huskey et Crawford, 1967; Prats, 1972; Asfari et Witherspoon, 1973). On a aussi utilisé l'approche basée sur des fractures discrètes pour analyser l'effet des conductances distribuées de fractures (Parsons, 1966) et des tailles distribuées de fractures (Caldwell, 1972) sur le tenseur de perméabilité des roches diaclasées. Enfin, on a démontré sur le terrain et en laboratoire l'effet important des contraintes sur l'écoulement du fluide dans les fractures, et on a utilisé des modèles numériques de couplage des contraintes et de l'écoulement (Gale, 1977; Noorishad et coll., 1982).

Actuellement, on consacre beaucoup d'efforts à l'étude de la migration des polluants existants et des polluants potentiels à partir des sites d'élimination de déchets toxiques et des dépôts des déchets radioactifs, particulièrement dans la roche fracturée. Même si la perméabilité d'une masse rocheuse fracturée est un paramètre important dans l'hydrogéologie des contaminants, d'autres paramètres hydrauliques, par exemple la porosité de fracture efficace (Gale et Rouleau, 1984) et la vitesse d'écoulement peuvent être plus importants car ils permettent de caractériser directement la vitesse de migration des polluants dissous dans l'eau souterraine.

De toute évidence, pour évaluer la porosité et la vitesse d'écoulement dans des roches fracturées, il faut avoir une description détaillée de la configuration du système de fractures. En général,

les fractures de la roche sont regroupées en familles d'après leur orientation. A l'intérieur d'une famille de fractures, il y a variabilité de l'orientation, de l'espacement et de la taille des fractures. Ces paramètres distribués doivent être caractérisés statistiquement (Baecher, 1983; Rouleau et Gale, 1985).

De même, il y a lieu d'adopter une approche stochastique pour intégrer ces paramètres de configuration dans des modèles numériques de réseaux de fractures (Conrad et Jacquin, 1973; Bridges, 1975; Veneziano, 1978; Hudson et Priest, 1979; Baczynski, 1980; LaPointe et Hudson, 1985). Des modèles de fractures discrètes stochastiques (FDS) comprenant une composante d'écoulement de fluide ont été élaborés d'abord dans des domaines bidimensionnels (Hudson et LaPointe, 1980; Long et coll., 1982; Robinson, 1982a; Samaniego et Priest, 1984; Andersson et coll., 1984) et, plus récemment, avec des réseaux de fractures tridimensionnels (Dershowitz et coll., 1985; Long et coll., 1985). On a aussi étudié le déplacement des solutés dans des réseaux de FDS orthogonales de taille distribuée et de position semi-aléatoire (Schwartz et coll., 1983). On traite de façon plus approfondie d'un grand nombre de ces études de référence et d'autres études dans les parties appropriées du texte.

Des études paramétriques, sans référence à un site particulier, peuvent indiquer l'effet relatif théorique des différents paramètres de configuration des fractures sur les variables dépendantes choisies, par exemple le débit total (Long et coll., 1982), la vitesse d'écoulement ou la distribution des temps de parcours des particules (Smith et Schwartz, 1984). Cependant, les simulations numériques propres à un site dans lesquelles on utilise des données de terrain réelles imposent des contraintes particulières à une étude. Les simulations propres à un site nécessitent des données de terrain détaillées et précises. De plus, il est évident que des comparaisons réalistes entre les données de terrain et les résultats du modèle numérique ne peuvent être faites que s'il y a cohérence des définitions des paramètres dans les données et dans le modèle.

Gale et Rouleau (1984) ont expliqué les principes de l'approche basée sur des FDS dans une étude propre à un site, dans le cadre d'une caractérisation hydrogéologique détaillée d'un dépôt de déchets dans de la roche fracturée ayant une perméabilité matricielle négligeable. Rouleau et Gale (1987) ont décrit une simulation par FDS de l'écoulement de l'eau souterraine dans une galerie expérimentale creusée dans du granite fracturé à Stripa (Suède).

Le présent rapport décrit les deux programmes utilisés par Rouleau et Gale (1987), NETWRK et NETFLO, et deux autres programmes connexes, APEGEN et NETRANS. Le programme NETWRK utilise la méthode de Monte Carlo pour générer des réseaux bidimensionnels de fractures discrètes basés sur les statistiques des principaux paramètres de configuration des fractures : densité, orientation, longueur de trace et ouverture des fractures. Le programme APEGEN permet ensuite de générer différentes distributions d'ouvertures à partir de la

configuration du réseau de fractures généré par le programme NETWRK. Le programme NETFLO calcule les paramètres de l'écoulement de fluide en régime permanent dans le réseau généré, pour des conditions aux limites particulières; il calcule également les valeurs cumulatives d'un certain nombre de paramètres d'écoulement dans chaque intervalle de direction de dix degrés. Ces derniers paramètres d'écoulement directionnels peuvent ensuite être utilisés par le programme NETRANS, qui utilise aussi une méthode de Monte Carlo, pour calculer le temps de parcours des particules sur une distance arbitraire (Rouleau, 1987).

J'ai adopté une approche modulaire dans l'écriture des programmes. De cette façon, le progiciel offre une plus grande souplesse pour l'addition de nouvelles routines ou la modification des routines existantes. De plus, il est ainsi possible d'analyser le réseau généré par un programme avant d'utiliser ce réseau dans un autre programme.

Les programmes ont été conçus en fonction d'applications propres à un site. Par exemple, les données d'entrée sur la configuration des fractures peuvent être recueillies directement sur place (Rouleau et Gale, 1985). Dans d'autres programmes de simulation de FDS, le nombre de fractures a généralement été utilisé comme l'un des paramètres d'entrée (Long et coll., 1982; Robinson, 1982a; Schwartz et coll., 1983; Samaniego et Priest, 1984; Smith et Schwartz, 1984). Dans notre traitement, nous utilisons à la place du nombre de fractures la densité de fractures (longueur totale des traces de fractures par unité de surface) comme paramètre d'entrée, étant donné que les données de trous de sonde permettent généralement une assez bonne estimation de ce dernier paramètre, indépendamment de la taille des fractures.

Les autres particularités de ce progiciel de simulations de FDS sont notamment l'optimisation du numérotage des noeuds, le stockage en tableau linéaire de la matrice des conductances aux noeuds et la méthode stochastique de deuxième degré utilisée pour l'étude de la migration des particules.

Simulation stochastique de systèmes de fractures: programmes NETWRK et APEGEN

2.1 Introduction

Au cours de la dernière décennie, certains chercheurs ont utilisé des méthodes numériques pour simuler les systèmes de fractures dans des masses rocheuses. Conrad et Jacquin (1973) ont proposé de produire des réseaux de lignes bidimentionnels pour étudier la configuration et la distribution des tailles de blocs rocheux. Hudson et Priest (1979) ont proposé une méthode tridimensionnelle pour résoudre le même problème. Bridges (1975) a proposé un modèle tridimensionnel basé sur une distribution aléatoire de disques dans l'espace combiné avec des graphiques de traces de fractures bidimensionnels en vue de «donner une image mentale des réseaux de fractures». Veneziano (1978) a proposé un autre modèle tridimensionnel basé sur des polygones de Poisson dans des plans aléatoires. Dans son modèle, Veneziano générait des plans infinis aléatoires dans l'espace. Il utilisait ensuite des lignes aléatoires dans chacun des plans pour définir des polygones convexes, dont certains représentaient des fractures discontinues. Le même auteur a étudié des applications possibles de ce modèle en évaluant les propriétés mécaniques d'une masse rocheuse. Dans des travaux de modélisation de la variabilité spatiale de la fracturation, Baczynski (1980) a produit des réseaux de lignes bidimensionnels servant à reproduire, pour une famille de fractures donnée, la distribution log-normale des espacements observée sur le terrain. LaPointe et Hudson (1985) ont aussi proposé des réseaux de lignes bidimensionnels générés par un traitement numérique en utilisant différents types de distributions pour les orientations et les longueurs des lignes. Plus récemment, Long et coll. (1982), Robinson (1982b) et Samaniego et Priest (1984) ont présenté les résultats de leurs travaux visant à générer des réseaux de lignes aléatoires bidimensionnels combinés avec le calcul numérique des paramètres d'écoulement d'un fluide en régime permanent (chapitre 3). A l'aide d'une approche numérique basée sur la théorie de la percolation élaborée dans le domaine de la physique théorique, Robinson (1982a) a trouvé les densités critiques pour des réseaux de fractures aléatoires bidimensionnels au-dessous desquelles il n'existe aucune voie d'écoulement continue dans un système de fractures.

Le programme de génération de réseaux de lignes NETWRK décrit dans le présent chapitre utilise une méthode de Monte Carlo pour générer un réseau de lignes de longueurs et orientations distribuées, de la même façon qu'on le fait dans les quatre dernières études susmentionnées. Dans ces modèles, chaque ligne représente la trace d'une fracture qui coupe une tranche de roche d'une unité d'épaisseur. Dans le programme NETWRK, les lignes sont générées une famille à la fois, à l'aide de paramètres de distribution distincts

pour les longueurs et les orientations de chaque famille. Des ouvertures de fractures sont aussi choisies à partir d'une distribution donnée et assignées à chaque ligne. On crée ainsi des fichiers de sortie qui peuvent être utilisés directement par d'autres programmes, par exemple pour calculer les paramètres de l'écoulement dans des réseaux de fractures bidimensionnels.

La création d'un modèle bidimensionnel constitue une première étape de la simulation numérique de réseaux de fractures discrètes. A partir d'un tel modèle, on peut évaluer les effets des fractures sur les propriétés mécaniques d'une masse rocheuse ou les caractéristiques de l'écoulement dans cette masse. Un modèle tridimensionnel permettrait une représentation plus réaliste d'un système de fractures, mais il n'existe pour l'instant aucun modèle tridimensionnel numérique pour les gros systèmes de fractures discrètes (c'est-à-dire en contenant des milliers) dont les tailles et les orientations sont distribuées. De plus, un modèle tridimensionnel nécessiterait beaucoup plus de mémoire d'ordinateur et de temps de calcul qu'un modèle bidimensionnel, pour le même nombre de fractures simulées.

2.2 Programme NETWRK : routine principale et données d'entrée

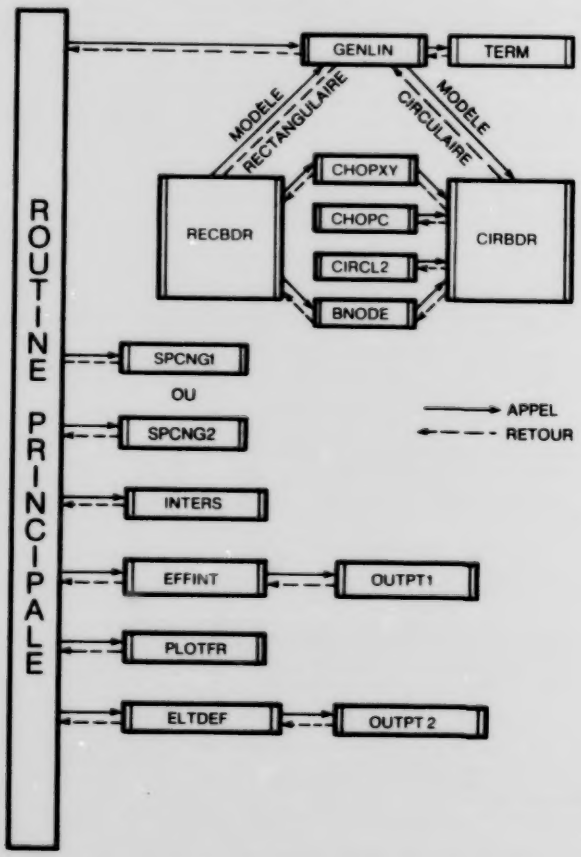


Figure 2.1. Structure du programme NETWRK.

La figure 2.1 montre la structure du programme NETWRK. Ce programme effectue la séquence d'opérations suivantes (figure 2.2) : 1) lecture des données d'entrée, 2) génération d'un réseau de lignes, 3) calcul de valeurs d'espacement (facultatif), 4) repérage de toutes les intersections «efficaces» du réseau, (c'est-à-dire des intersections qui font partie d'une voie d'écoulement continue),

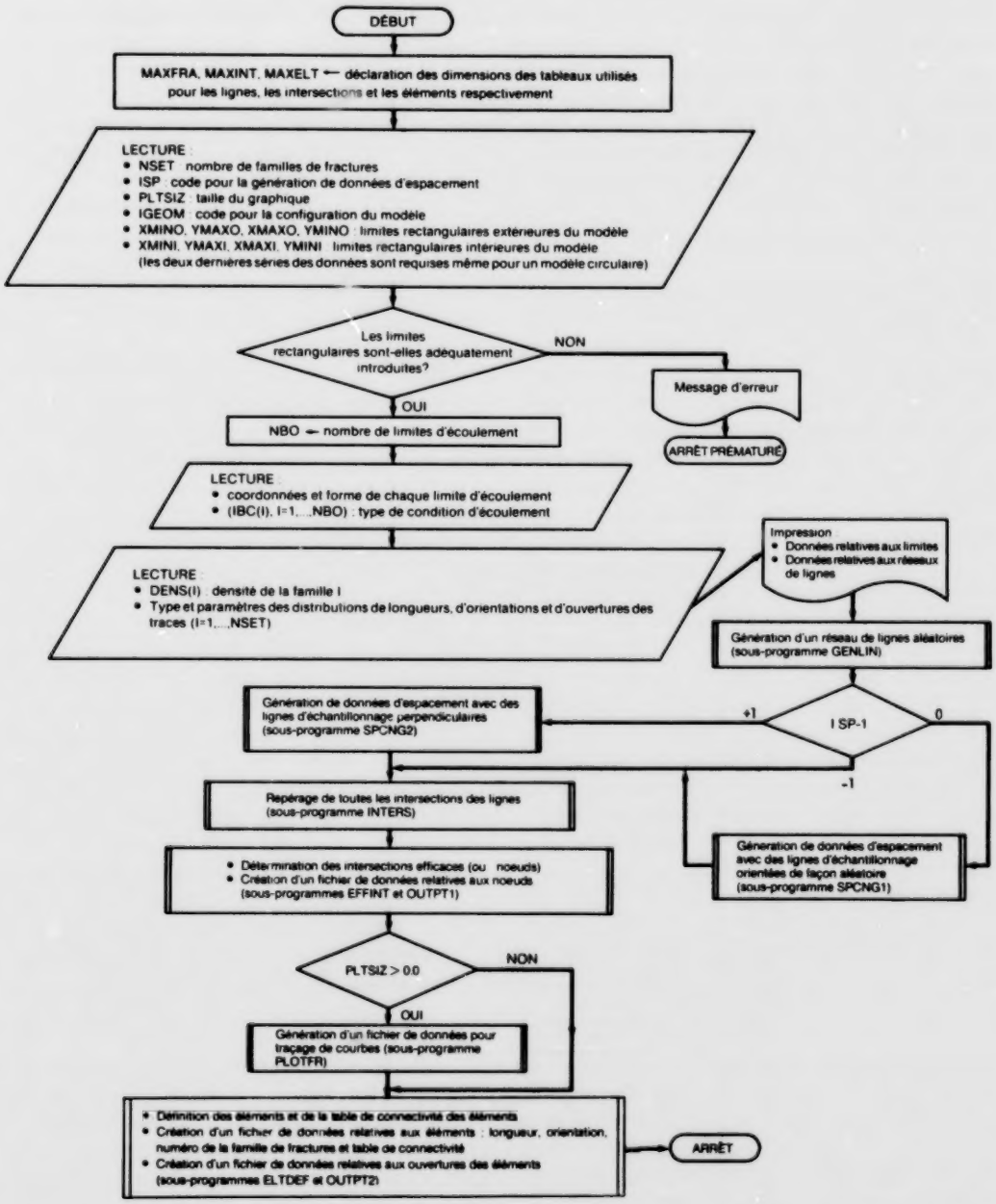


Figure 2.2. Organigramme de la routine principale du programme NETWRK.

5) génération d'un fichier de données pour traçage de courbes (facultatif), 6) définition des «éléments» (c'est-à-dire chaque segment de ligne entre deux intersections efficaces consécutives) et enregistrement des numéros des noeuds qui identifient chaque élément.

Les données d'entrée du programme sont principalement de deux types : 1) données décrivant la configuration du réseau de fractures 2) données relatives à la configuration des limites (forme et coordonnées). Le réseau de fractures est défini par la densité de chaque famille de fractures (DENS [L^{-1}]) et par le type et les paramètres de la distribution des longueurs de traces, des orientations et des ouvertures des fractures pour chaque famille. Le programme NETWRK peut traiter deux formes de modèles : des modèles rectangulaires et des modèles circulaires (figure 2.3; aussi annexe D). Dans un modèle circulaire, un seul quadrant d'un anneau complet, défini par deux cercles concentriques et deux rayons, est représenté. Chaque modèle nécessite une limite extérieure et une limite intérieure. Les lignes du réseau sont générées pour le modèle entier à l'intérieur des limites extérieures. Les limites intérieures servent à découper le réseau sur tous les côtés et elles sont les seules limites considérées dans le programme NETFLO (chapitre 3). La marge tampon entre les limites intérieures et extérieures élimine la plus grande partie de l'effet de bord attendu dans un réseau de lignes généré en raison de la densité systématiquement moins grande des fractures à proximité des limites.

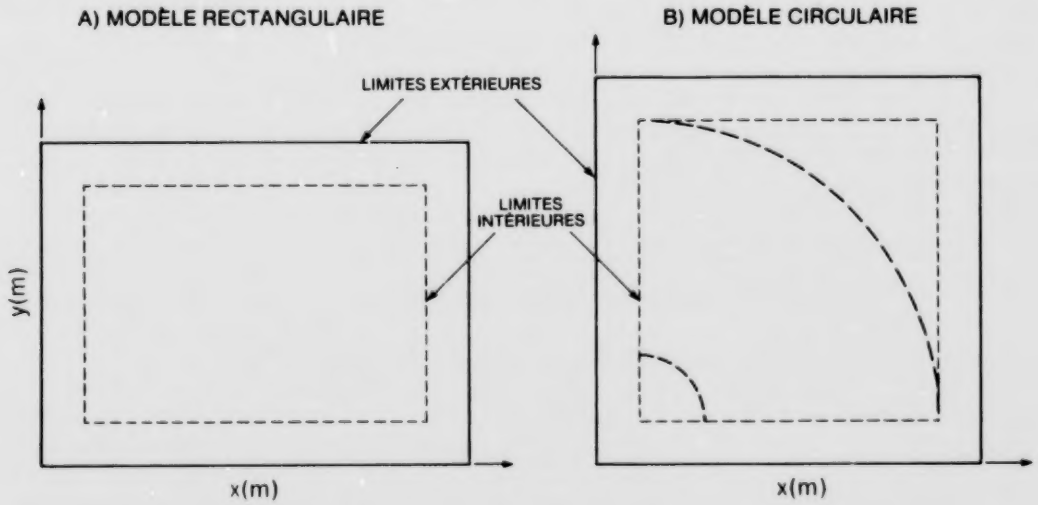


Figure 2.3. Configuration du modèle : A) modèle rectangulaire, B) modèle circulaire.

2.3 Génération du réseau de lignes

La génération du réseau de lignes est effectuée par le sous-programme GENLIN (figure 2.4). A cette étape du traitement, l'ordinateur calcule d'abord la surface du modèle délimitée par les limites intérieures. Le réseau de lignes est ensuite généré une

famille à la fois. La longueur totale désirée (DESLEN) des lignes est calculée et une estimation est faite du nombre de lignes à générer (ESTNFR). Les variables DESLEN et ESTNFR sont utilisées comme paramètres de contrôle dans le processus de génération des lignes. La valeur de la variable ESTNFR est déterminée en multipliant un facteur COEF donné (un paramètre d'entrée généralement compris entre 2 et 5) par l'estimation initiale du nombre de lignes basée sur la longueur moyenne d'une trace et la valeur de DESLEN. Cette valeur de ESTNFR est plus grande que la valeur généralement requise, mais on est ainsi assuré que la densité de fractures désirée sera obtenue dans tous les

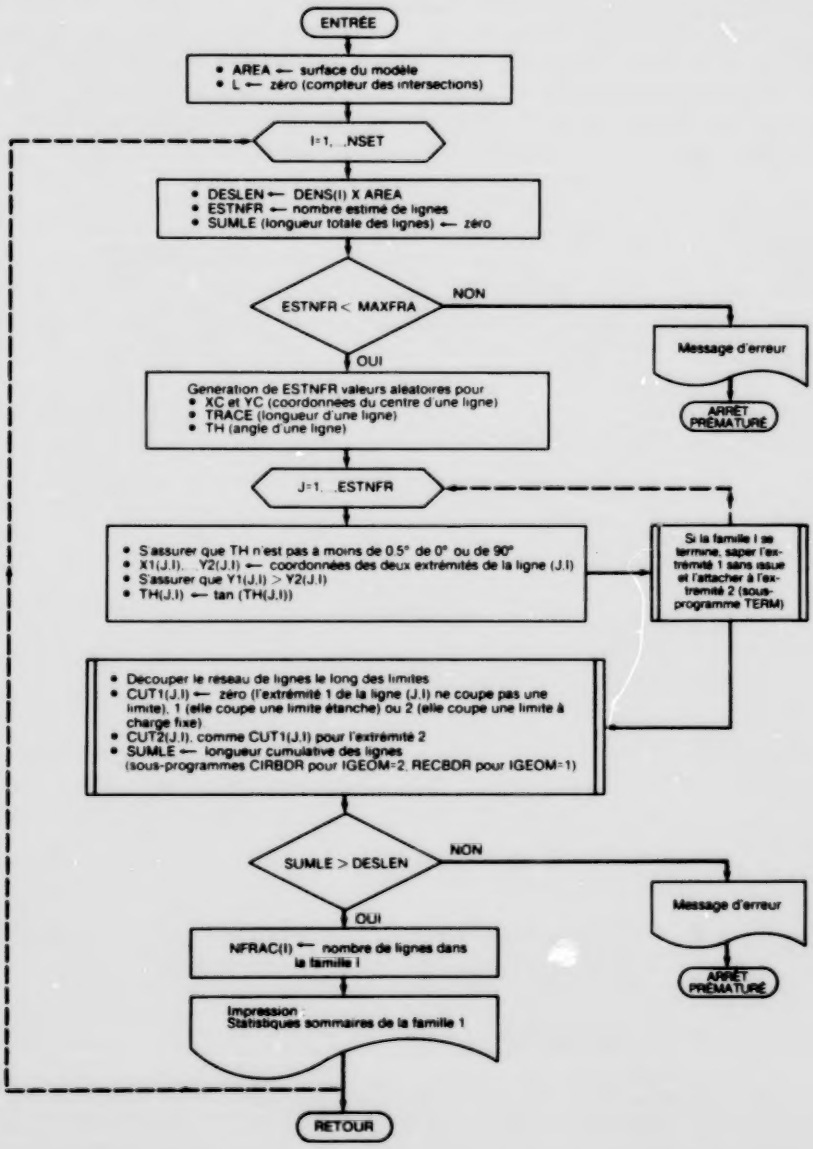


Figure 2.4. Organigramme du sous-programme GENLIN.

réseaux générés. Dans certains cas, les lignes supplémentaires qui sont générées compensent exactement la proportion élevée de traces courtes apparaissant au début du processus de génération des longueurs de traces et le nombre élevé de traces dans la marge entre la limite extérieure et la limite intérieure, comme on l'explique ci-après.

Ensuite, un nombre fixe ESTNFR de points aléatoires sont générés à l'intérieur de la limite extérieure, chaque point correspondant au centre d'une ligne (XC, YC). Par la suite, le sous-programme génère le même nombre 1) de longueurs de lignes (TRACE, en mètres) à l'aide soit d'une valeur unique, soit d'une distribution log-normale, 2) d'orientations de lignes (TH, en degrés) à l'aide soit d'une valeur unique, soit d'une distribution normale. Les éléments des tableaux XC, YC, TRACE et TH sont utilisés concurremment par la suite dans le programme pour calculer les coordonnées des deux extrémités de chaque ligne (X1, Y1 et X2, Y2).

Pour le calcul des coordonnées des extrémités des lignes, on adopte deux conventions visant à simplifier les opérations dans les autres sous-programmes. La première convention consiste à considérer toute ligne s'écartant d'une valeur inférieure à 0.5 degré à partir des orientations 0.0, 90.0, 180.0 ou 270.0 comme ayant une orientation de -0.5, +0.5, 89.5 ou 90.5. Dans le cas des lignes quasi-parallèles à l'axe des x (figure 2.3), cette convention assure qu'il n'y aura aucune valeur indéterminée dans la définition des éléments de lignes (section 2.7). En réalité, les éléments suivant une ligne sont définis après le tri (par ordre croissant de la valeur de y) de toutes les intersections le long de cette ligne. Dans le cas des lignes quasi-parallèles à l'axe des y (figure 2.3), la même convention fixe une limite supérieure à la valeur de la tangente des angles des lignes. Elle prévient ainsi les erreurs en virgule flottante, particulièrement dans le calcul des intersections de lignes (section 2.5). La deuxième convention assure simplement que l'extrémité 1 (c'est-à-dire X1, Y1) est l'extrémité ayant la plus grande valeur de Y. Cette convention simplifie la programmation, en particulier pour le découpage des lignes le long des limites intérieures, opération qui est décrite brièvement ci-après.

A ce stade, le sous-programme TERM peut être appelé (lorsque la valeur 1 est assignée au paramètre d'entrée ITERM) pour faire terminer les fractures de la famille générée sur des fractures de familles générées antérieurement. Cette opération vise à simuler de façon plus réaliste les systèmes naturels qui sont souvent composés de fractures se terminant sur d'autres fractures. Le sous-programme TERM coupe le segment d'extrémité au-dessus de la plus haute intersection (par rapport à l'axe des y) le long d'une fracture; il reporte ensuite ce segment à l'extrémité inférieure de la même fracture, de façon que la longueur totale de la fracture reste la même. De cette façon, une seule extrémité d'une fracture donnée se termine sur une autre fracture. De plus, les fractures d'une famille donnée peuvent seulement se terminer sur des fractures de familles qui ont été générées antérieurement. De toute évidence, les fractures de la famille 1 ne peuvent pas se terminer sur d'autres fractures. Un

exemple de ce procédé de coupure est donné dans la réalisation 992 décrite à l'annexe F.

Le réseau de lignes est découpé le long des limites intérieures par le sous-programme RECBDR lorsque le modèle est rectangulaire ou par le sous-programme CIRBDR lorsque le modèle est circulaire. Ces deux sous-programmes et leurs sous-programmes subordonnés rejettent toute ligne située en entier hors des limites intérieures, coupent toutes les lignes traversant une limite intérieure, calculent une longueur corrigée pour ces lignes et calculent les nouvelles coordonnées des extrémités découpées. Toute intersection entre une ligne et une limite intérieure est aussi définie comme un noeud limite. Les coordonnées de chaque noeud sont enregistrées dans les tableaux XORD et YORD, et le tableau IB identifie la limite à laquelle appartient un noeud. Les sous-programmes RECBDR et CIRBDR totalisent aussi progressivement pour chaque famille la longueur totale (SUMLE) des segments de lignes délimités par les limites intérieures. Ces sous-programmes repassent la commande au générateur de lignes GENLIN lorsque la variable SUMLE atteint la longueur totale désirée (DESLEN) pour la famille considérée. Le nombre de lignes considéré pour chaque famille est ensuite enregistré dans NFRAC.

2.4 Génération de données d'espacement

Comme le montre l'organisme de la routine principale (figure 2.2), après l'étape de génération des lignes pour toutes les familles, le programme peut générer un fichier de données d'espacement facultatif. Le nombre de valeurs d'espacement générées pour chaque famille de fractures est fixé arbitrairement à 150. L'espacement se définit comme le produit de la distance entre deux intersections consécutives d'une ligne d'échantillonnage avec des fractures de la même famille par le cosinus de l'angle entre la ligne d'échantillonnage et la normale au plan moyen de cette famille. L'utilisateur peut choisir entre deux sous-programmes pour la génération des données d'espacement selon le type de processus d'échantillonnage désiré. Le sous-programme SPCNG1 utilise seulement des lignes d'échantillonnage perpendiculaires à la direction moyenne d'une famille. Le sous-programme SPCNG2 utilise des lignes d'échantillonnage orientées de façon aléatoire dans le plan du modèle. L'analyse statistique des données d'espacement générées peut être utilisée pour des comparaisons avec les données d'espacement sur le terrain (Rouleau et Gale, 1985).

2.5 Intersections de lignes

Lorsque toutes les lignes ont été générées, la routine principale appelle les sous-programmes INTERS et EFFINT pour repérer toutes les intersections de lignes efficaces dans le réseau.

On peut calculer, à l'aide de relations trigonométriques simples, les coordonnées du point d'intersection de deux lignes dans un plan lorsqu'on connaît les coordonnées d'un point et l'orientation

de chaque ligne. Pour calculer l'ordonnée, on peut utiliser la relation :

$$YI = \frac{Y1(1)}{\tan TH(1)} - \frac{Y1(2)}{\tan TH(2)} - X1(1) + X1(2) \times \frac{\tan TH(1) \times \tan TH(2)}{\tan TH(2) - \tan TH(1)} \quad (2.1)$$

où YI est l'ordonnée du point d'intersection, X1 (1 et 2) et Y1 (1 et 2) sont les coordonnées d'un point connu des lignes 1 et 2 (p. ex., extrémité 1 des lignes du réseau), TH (1 et 2) est l'angle entre les lignes (1 et 2) et l'axe des x. L'abscisse du point d'intersection, XI, peut ensuite être calculée à l'aide de la relation:

$$XI = \frac{YI - Y1(1)}{\tan TH(1)} + X1(1) \quad (2.2)$$

Le sous-programme INTERS (figure 2.5) se compose essentiellement de quatre boucles DO imbriquées. Pour chaque ligne, l'ordinateur explore toutes les autres lignes du réseau, mais des mesures sont prises pour faire en sorte que la recherche de l'intersection entre les deux mêmes lignes soit effectuée une seule fois. D'abord, l'ordonnée (YI) d'un point d'intersection possible entre deux lignes est calculée à l'aide de l'équation (2.1). Lorsque cette valeur de YI appartient aux valeurs possibles des deux lignes dans la direction des y, il existe un point d'intersection, et on calcule son abscisse à l'aide de l'équation (2.2). Les coordonnées du point d'intersection sont enregistrées dans les tableaux XORD et YORD. Les numéros des lignes formant une intersection sont enregistrés dans les tableaux INC1 et INC2.

Une fois que le sous-programme INTERS a repassé la commande à la routine principale, le sous-programme EFFINT est appelé pour marquer la plupart des intersections non efficaces, c'est-à-dire les intersections qui n'appartiennent pas à une voie d'écoulement continue entre deux limites à charge fixe. Le sous-programme identifie ces intersections non efficaces en marquant systématiquement toute intersection qui est la seule intersection sur une ligne. Les intersections efficaces sont aussi appelées noeuds dans le texte. Il ne faut toutefois pas oublier que les intersections non efficaces ne sont pas nécessairement toutes marquées avec cette façon de procéder. En effet, des boucles isolées formées par trois intersections ou plus existent parfois dans un réseau, et dans ce cas, aucune de ces intersections n'est marquée comme non efficace.

2.6 Génération d'un fichier de données pour le traçage de courbes

A ce stade, le sous-programme PLOTFR est appelé (sauf si on a donné une valeur négative au paramètre d'entrée PLTSIZ) pour créer un fichier de données servant au traçage du réseau de lignes au moyen d'un traceur numérique. Le graphique résultant représente toutes les limites et le réseau de lignes après découpage suivant les limites intérieures. Les intersections efficaces sont identifiées par un petit cercle. Les numéros de lignes et les numéros d'intersections sont aussi représentés sur un graphique comprenant moins de 200

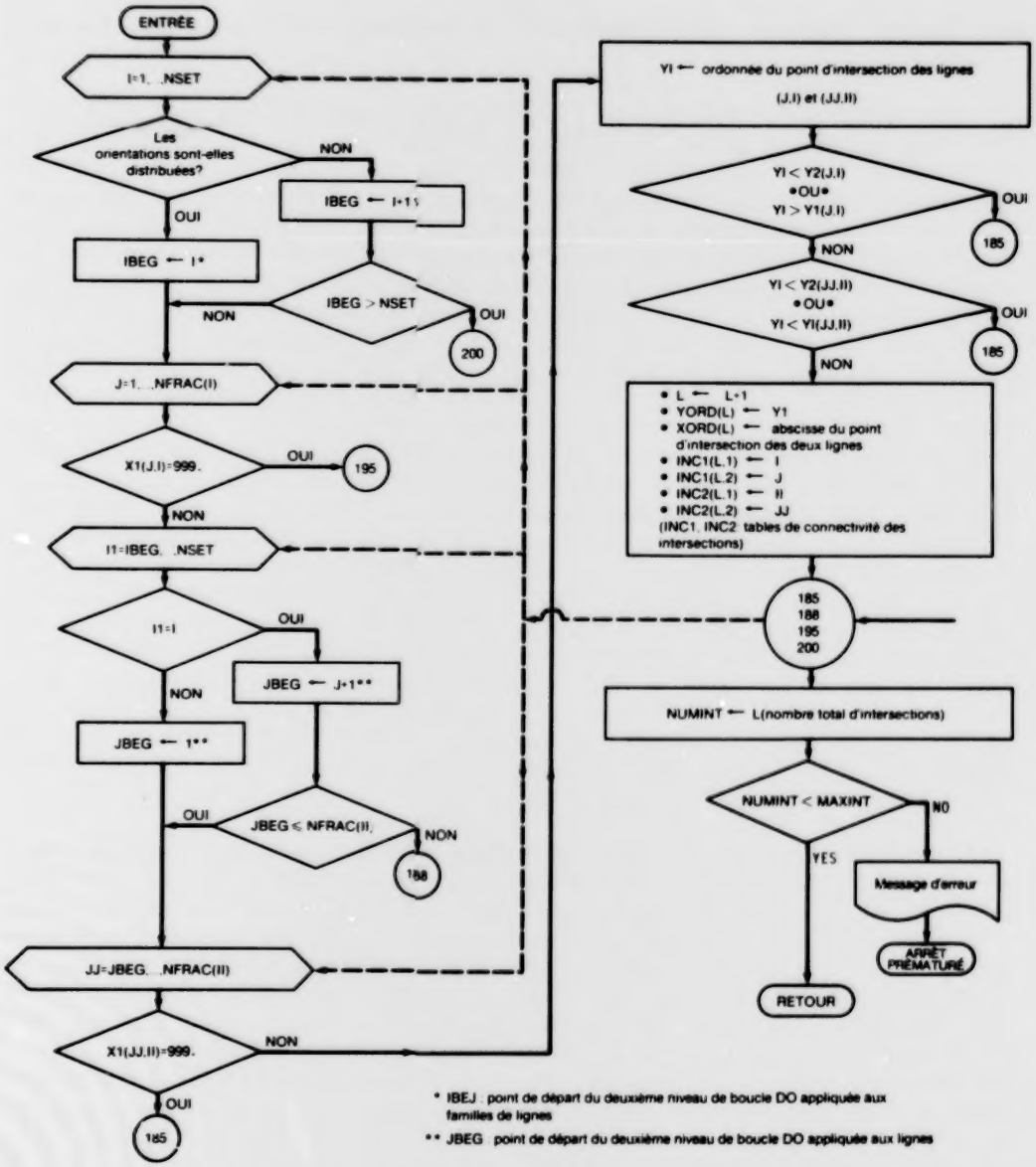


Figure 2.5. Organigramme du sous-programme INTERS.

intersections efficaces, lorsque la valeur du paramètre d'entrée IPRT est supérieure à 1.

2.7 Détermination des éléments de lignes

La dernière étape du programme NETWRK est réalisée par le sous-programme ELTDEF (figure 2.6). Au moyen de trois boucles DO imbriquées, le sous-programme ELTDEF identifie d'abord toutes les

intersections efficaces d'une ligne donnée en utilisant les tables de connectivité des intersections INC1 et INC2. Après le classement de ces intersections en fonction des valeurs croissantes de y, tous les éléments ainsi définis sont identifiés et leur table de connectivité est enregistrée dans les vecteurs NOD1 et NOD2. Au cours du même traitement, le sous-programme ELTDEF assigne une «ouverture de fracture» aléatoire à chaque ligne, à partir soit d'une valeur unique, soit d'une distribution log-normale selon les paramètres d'entrée. Le sous-programme ELTDEF élimine aussi les segments sans

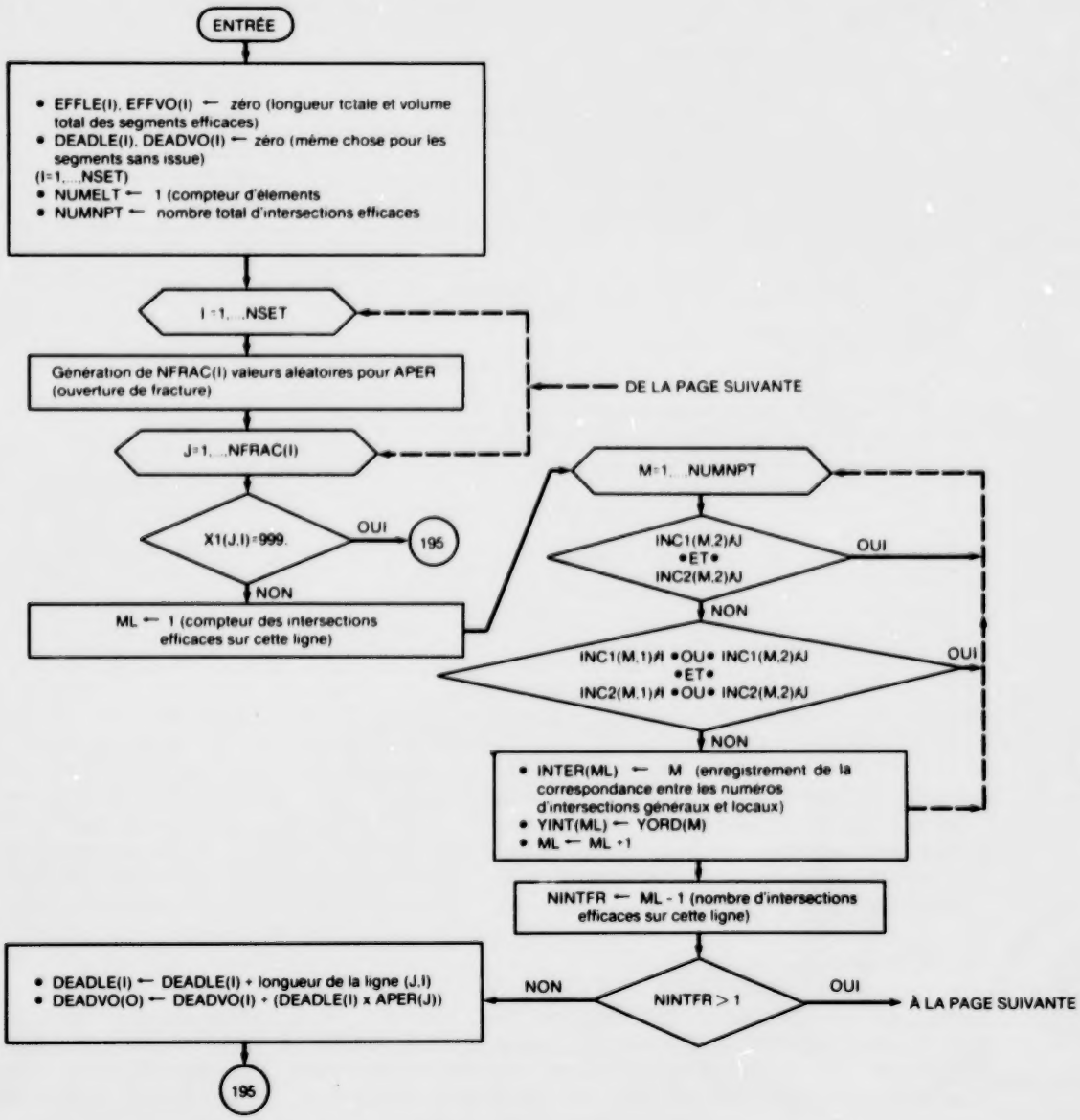


Figure 2.6. Organigramme du sous-programme ELTDEF.

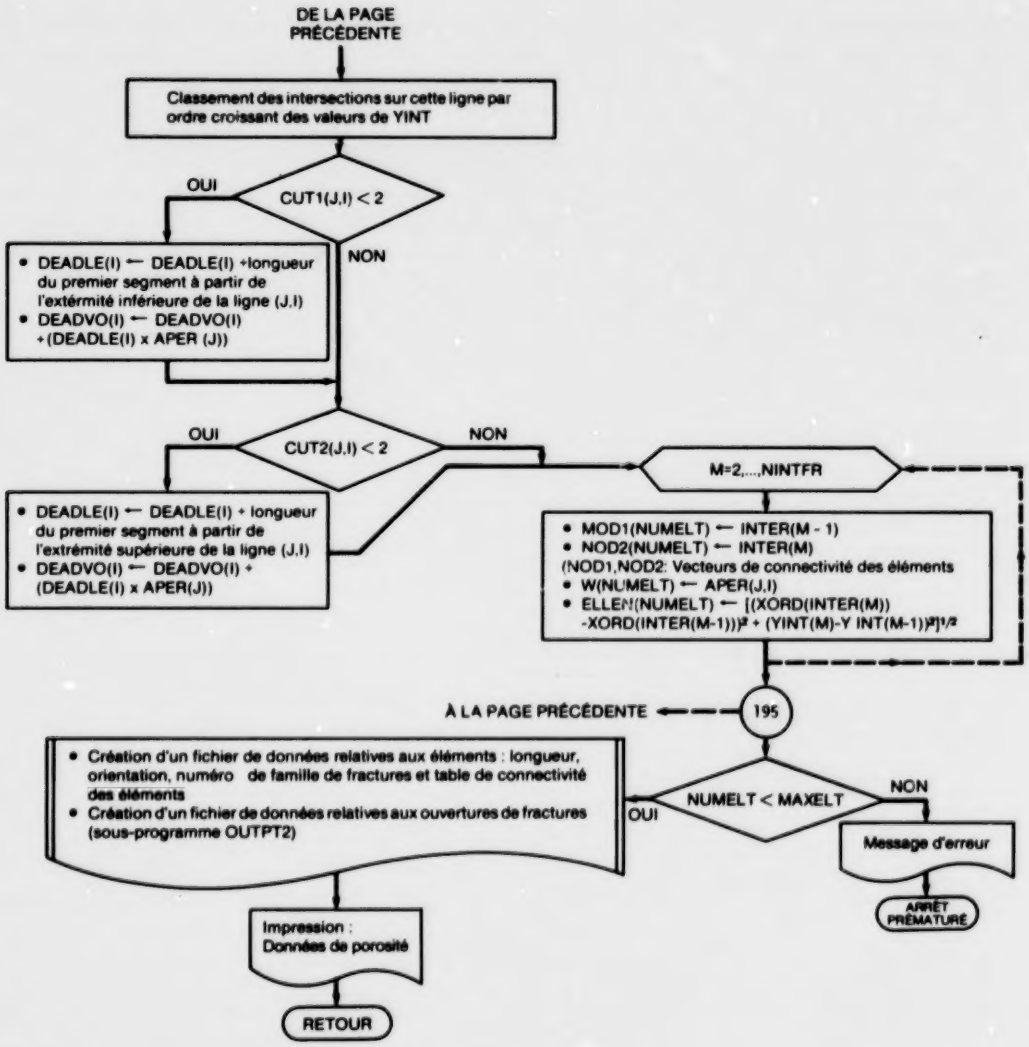
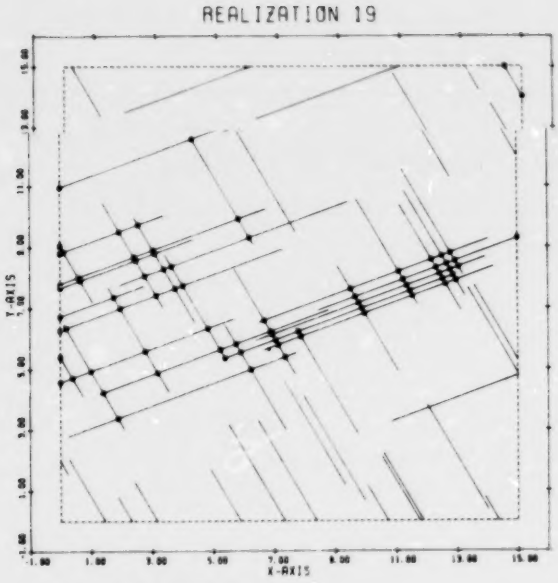


Figure 2.6. (suite).

issue, comprenant les lignes complètes et les extrémités de lignes, après avoir enregistré la longueur totale et le «volume» total de chaque famille. L'exécution du sous-programme ELTDEF est répétée pour chaque distribution d'ouvertures différente pour une réalisation particulière d'un réseau de lignes.

2.8 Exemple simple

La figure 2.7 montre un réseau de lignes généré par le programme NETWRK. Dans cet exemple simple, on a utilisé des valeurs uniques pour les longueurs et orientations des lignes, et pour les ouvertures des fractures. On remarque aussi que le paramètre d'entrée IPRT avait la valeur 1; par conséquent, les numéros de lignes et



PARAMÈTRES D'ENTRÉE

	FAMILLE 1	FAMILLE 2
DENSITÉ (m^{-1})	0.5	0.5
ORIENTATION (d)	20	120
LONGUEUR DE TRACE (m)	8	4
OUVERTURE (m)	20×10^{-6}	20×10^{-6}

Figure 2.7. Réseau de lignes simples généré par le programme NETWRK.

d'intersections ne sont pas inscrits sur le graphique de la figure 2.7 produit par ordinateur. De plus, le paramètre de contrôle de fin de ligne ITERM a été fixé à zéro pour les deux familles, de sorte qu'aucune ligne ne se termine sur une autre ligne. Le tableau 2.1 montre certains résultats choisis dans les données de sortie pour cet exemple. D'autres données de sortie pouvant être obtenues en option mais non représentées dans le tableau 2.1 sont notamment : 1) une liste de toutes les lignes de chaque famille, avec leur longueur, leur orientation et les coordonnées de leurs extrémités; 2) une liste de toutes les intersections avec leurs coordonnées et les numéros de lignes qui se coupent; 3) une liste de tous les éléments avec les numéros des noeuds, les numéros de lignes auxquelles ils appartiennent, leur longueur et leur ouverture de fracture correspondante. L'annexe B donne une liste des différents fichiers de sortie et l'annexe F donne des exemples plus détaillés.

L'exemple simple de la figure 2.7 fait déjà ressortir un aspect intéressant d'un réseau de fractures discontinues : pour une même densité de fractures, la proportion des segments de fractures sans issue est plus élevée dans le cas d'une famille de fractures courtes (75 % pour la famille 2 de cet exemple) que dans le cas d'une famille de fractures longues (34 % pour la famille 1).

Tableau 2.1. Résultats choisis pour le réseau de lignes de la figure 2.7

Nombre total d'intersections (NUMINT):		97
Nombre d'intersections efficaces (NUMNPT):		76
Nombre d'éléments (NUMELT):		108
	Famille 1	Famille 2
DESLEN (m)	112.5	112.5
SUMLE (m)	114.8	113.8
NFRAC	18	39
EFFLE (m)	76.4	28.4
DEADLE (m)	38.4	85.5
EFFVO (m^3)	1.5×10^{-3}	5.7×10^{-4}
DEADVO (m^3)	7.7×10^{-4}	1.7×10^{-3}

2.9 Génération de distributions d'ouvertures supplémentaires : programme APEGEN

Dans le sous-programme ELTDEF (section 2.7), des valeurs d'ouvertures distribuées sont générées et une valeur d'ouverture est assignée à chaque fracture générée. Un deuxième programme, APEGEN (figure 2.8), permet aussi de générer différentes valeurs d'ouvertures. Le programme APEGEN permet d'utiliser le même réseau de lignes généré autant de fois qu'on le désire, avec différents paramètres de distribution des ouvertures.

Le programme APEGEN a en plus l'avantage de donner différentes valeurs d'ouvertures aux différents éléments suivant les mêmes fractures. Enfin, avec seulement quelques ajouts mineurs, le programme APEGEN pourrait calculer des valeurs d'ouvertures qui sont fonction de l'orientation ou de la position des éléments de fractures.

2.10 Conclusion

Le programme NETWRK est un outil numérique qu'on peut utiliser dans un certain nombre d'applications. A lui seul, il fournit des renseignements sur la topologie d'un système de fractures (p. ex., degré d'intercommunication, rapport de la porosité des fractures connectées à la porosité totale, et distribution d'espacement). Le programme NETWRK peut être utilisé conjointement

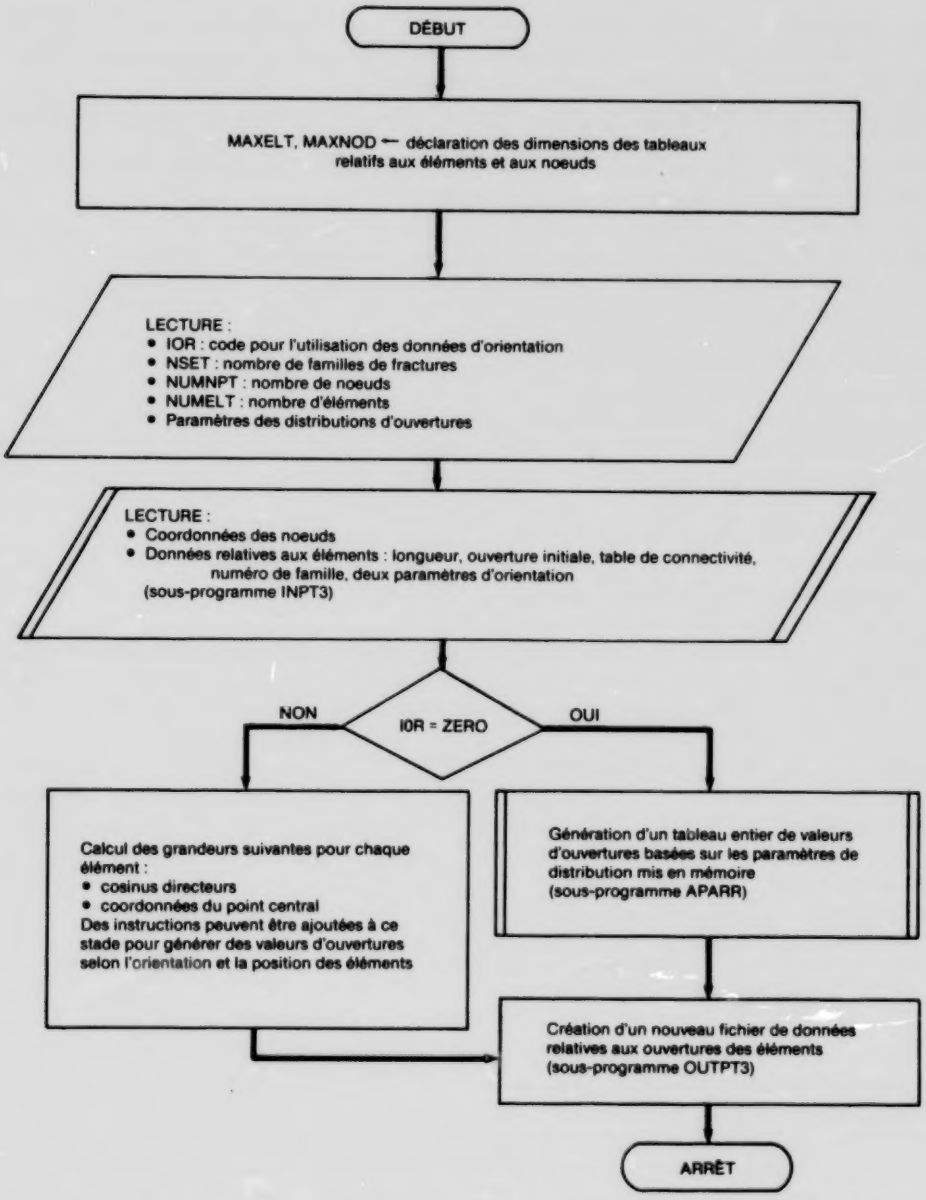


Figure 2.8. Organigramme de la routine principale du programme APEGEN.

avec d'autres programmes pour évaluer le comportement mécanique d'une masse rocheuse fracturée ou pour calculer les paramètres de l'écoulement de fluide dans un réseau de fractures. Le prochain chapitre décrit le programme NETFLO, qui calcule les paramètres de l'écoulement de fluide en régime permanent dans un réseau de fractures généré par le programme NETWRK, pour des conditions aux limites particulières.

Simulation numérique de l'écoulement de fluide dans un réseau bidimensionnel de fractures discrètes: programme NETFLO

3.1 Introduction

Le présent chapitre traite du programme de simulation numérique NETFLO qui simule l'écoulement de fluide dans des réseaux de fractures discrètes comme ceux que génère le programme NETWRK (chapitre 2). On a écrit le programme NETFLO en vue d'évaluer le rôle de la configuration des fractures dans la détermination des caractéristiques d'écoulement de l'eau souterraine dans des masses rocheuses à matrice de faible perméabilité.

Des études antérieures proposaient une variété de techniques pour simuler l'écoulement de fluide dans des fractures discrètes. Ollos (1963) a construit un modèle physique à l'aide de tuyaux de PVC communiquant entre eux à angle droit en vue d'étudier les caractéristiques d'écoulement de l'eau souterraine d'un système de fractures. Huskey et Crawford (1967) ont décrit un modèle analogique électriques consistant en un bain électrolytique et des bandes de cuivre simulant les fractures. Wittke (1970) a utilisé un autre modèle analogique électrique composé d'un grand nombre de résistances électriques disposées en réseaux. Caldwell (1972) a utilisé un troisième type de modèle analogique électrique réalisé à l'aide de papier conducteur comprenant des perforations pour simuler les blocs rocheux perméables. La technique de Caldwell est l'image inverse d'une technique décrite par Marcus et Evenson (1961) pour simuler l'effet d'agencements réguliers de couches imperméables minces dans une matrice perméable. Hudson et LaPointe (1980) ont proposé un modèle analogique électrique consistant en un circuit imprimé qui était une réplique d'un réseau de traces de diaclases aléatoires généré par ordinateur. Des méthodes numériques mises aussi au point pour l'étude des problèmes relatifs aux réseaux de résistances électriques ont été adaptées à l'étude de l'écoulement de fluide dans des réseaux de fractures (Parsons, 1965; Louis, 1969). Castillo et coll. (1972) ont utilisé une méthode combinant une solution itérative et une solution directe pour résoudre le problème de l'écoulement en régime permanent dans des réseaux de fractures de configuration régulière. Avec cette dernière approche, il est possible d'étudier des phénomènes non linéaires résultant de la rugosité des fractures et de l'écoulement turbulent.

Wittke (1970) a utilisé la méthode par éléments finis pour résoudre un problème d'écoulement tridimensionnel de fluide en régime

permanent dans un réseau de fractures continues à l'intérieur d'une matrice imperméable. Wilson et Witherspoon (1970) ont élaboré un modèle à éléments finis bidimensionnel pour résoudre le même problème d'écoulement de fluide dans un réseau de fractures de configuration arbitraire. Le programme de modélisation de Wilson et Witherspoon est efficace sur le plan du temps de calcul mais non sur le plan de l'utilisation de mémoire d'ordinateur puisqu'il nécessite la mémorisation d'une matrice carrée complète d'ordre N, N désignant le nombre de noeuds. Ces chercheurs ont présenté des exemples de réseaux de configuration régulière entièrement connectés. Asfari et Witherspoon (1973) ont présenté les résultats d'un autre programme à éléments finis appliqué à des réseaux réguliers de fractures courtes sans aucune connexion dans une matrice perméable. Long et coll. (1982) ont adapté le programme de Wilson et Witherspoon à des réseaux bidimensionnels de fractures discontinues dans une matrice imperméable générés de façon aléatoire. Robinson (1982b), qui étudiait aussi des réseaux aléatoires, a utilisé une solution directe (élimination de Gauss) pour résoudre le problème de l'écoulement en régime permanent, tandis que Samaniego et Priest (1984) ont essayé deux méthodes différentes : une méthode itérative (relaxation) et une méthode directe (bi-factorisation). Plus récemment, on a élaboré des modèles numériques pour simuler l'écoulement de fluide en régime permanent dans des réseaux de fractures tridimensionnels. Dershowitz et coll. (1985) ont utilisé une approche par éléments finis; Long et coll. (1985) ont adopté une méthode analytique-numérique mixte basée sur des sources de lignes (puits); pour leur part, Shapiro et Andersson (1985) ont proposé une méthode basée sur les éléments limites. En théorie, un modèle tridimensionnel de l'écoulement dans un système de fractures est plus réaliste qu'un modèle bidimensionnel. Malheureusement, on ne dispose encore d'aucun modèle tridimensionnel pouvant simuler l'écoulement dans des réseaux comprenant des milliers d'intersections de fractures. De plus, pour le même espace de mémoire d'ordinateur et pour des logiciels similaires, on peut simuler un système de fractures beaucoup plus gros avec un modèle bidimensionnel qu'avec un modèle tridimensionnel.

La théorie de l'écoulement dans un réseau de fractures bidimensionnel utilisée dans cette étude ressemble beaucoup à la théorie de l'écoulement de fluide dans des réseaux de canalisations. Ollos (1963) a utilisé cette similarité dans des modèles physiques. On trouve dans la documentation une quantité considérable d'articles consacrés à la solution numérique de l'écoulement dans des réseaux de canalisations transportant par exemple de l'eau, du gaz naturel ou des produits du pétrole (Mah et Shacham, 1978). Normalement, un problème relatif à un réseau de canalisations a une structure générale prédéterminée, et on utilise des méthodes numériques pour l'analyse de sensibilité et l'optimisation du modèle. Etant donné que la même structure de problème se répète un grand nombre de fois pendant une étude, il est avantageux d'optimiser dès le début la structure des matrices à résoudre numériquement. Les méthodes basées sur la théorie linéaire et la théorie des graphes sont utilisées couramment pour l'analyse de ces types de réseaux de canalisations (Kesavan et Chandrashekar, 1972; Isaacs et Mills, 1980).

Le modèle numérique proposé dans cette étude est bidimensionnel, tandis que la matrice rocheuse est considérée imperméable. On crée d'abord une matrice des conductances aux noeuds en appliquant la contrainte d'équilibre des masses à chaque noeud du réseau (c'est-à-dire que la somme des débits à tout noeud libre doit être nulle). Etant donné qu'il est destiné à être appliqué à des problèmes réels sur le terrain, ce programme informatique doit simuler les fractures d'une tranche de roche assez grosse pour être considérée comme statistiquement homogène par rapport au système de fractures. De plus, le processus de génération de fractures aléatoires produit une matrice numérique de structure très clairsemée. Par un renumérotage systématique de tous les noeuds et par l'utilisation d'un mode de stockage à largeur de bande variable, le programme proposé réduit considérablement l'espace de mémoire requis pour une matrice clairsemée de grande taille. On utilise ensuite une méthode directe basée sur l'algorithme de Choleski pour résoudre l'ensemble d'équations d'une façon qui demande peu de temps de calcul.

3.2 Formulation du modèle

Dans le cas de l'écoulement laminaire en régime permanent d'un fluide visqueux et incompressible entre deux plaques parallèles lisses, la solution analytique de l'équation de Navier-Stokes donne la relation suivante pour le débit volumique par unité de largeur (p. ex., Huitt, 1956):

$$q = \frac{W^3 \gamma}{12\mu} \frac{\Delta\varphi}{\Delta x} \tag{3.1}$$

où W est l'espacement des plaques [L], γ est le poids volumique du fluide [F/L^3], μ est la viscosité dynamique [FT/L^2], x est la distance suivant les plaques [L] et φ est la charge hydraulique [L]. Dans la résolution des problèmes relatifs à l'écoulement dans des réseaux, il est souvent pratique de définir la conductance du conduit de fluide de longueur ℓ par :

$$e = \frac{W^3 \gamma}{12\mu\ell} \qquad [L^2\, T^{-1}] \tag{3.2}$$

c'est-à-dire la conductivité hydraulique multipliée par la section transversale et divisée par la longueur.

Considérons par exemple la figure 3.1. Le débit au noeud 1 est représenté par :

$$e_a(\varphi_2 - \varphi_1) = -q_a \tag{3.3}$$

Dans la convention adoptée, l'écoulement entrant dans le système est positif et l'écoulement en sortant est négatif. Pour le noeud interne 2, il faut, en vertu de la contrainte d'équilibre des masses, que la somme des débits entrants et des débits sortants de ce noeud soit nulle :

$$e_a(\varphi_1 - \varphi_2) + e_b(\varphi_3 - \varphi_2) + e_d(\varphi_4 - \varphi_2) = 0 \tag{3.4}$$

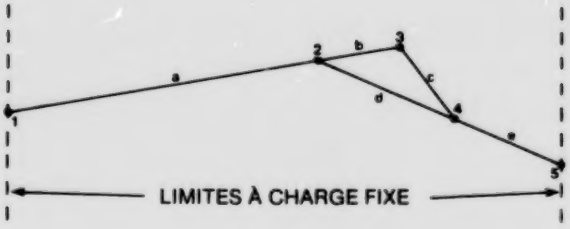


Figure 3.1. Réseau de fractures composé de cinq segments connectés.

Sous forme matricielle, les équations (3.3) et (3.4) du système complet de la Figure 3.1 donnent:

$$\begin{array}{c} \text{noeuds} \rightarrow \\ \downarrow \end{array} \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ e_{-a} & -e_a & 0 & 0 & 0 \\ -e_a & (e_a+e_b+e_d) & -e_b & -e_d & 0 \\ 0 & -e_b & (e_b+e_c) & -e_c & 0 \\ 0 & -e_d & -e_c & (e_c+e_d+e_a) & -e_e \\ 0 & 0 & 0 & -e_e & e_e \end{bmatrix} \times \begin{bmatrix} \varphi_1 \\ \varphi_2 \\ \varphi_3 \\ \varphi_4 \\ \varphi_5 \end{bmatrix} = \begin{bmatrix} q_1 \\ 0 \\ 0 \\ 0 \\ q_5 \end{bmatrix} \quad (3.5)$$

Les nombreuses méthodes permettant de résoudre simultanément des systèmes d'équations comme le système (3.5) peuvent être classées en méthodes directes et en méthodes itératives (p. ex., Jennings, 1977). Les méthodes directes sont généralement plus rapides mais elles demandent plus de mémoire d'ordinateur que les méthodes itératives. Par exemple, Wilson et Witherspoon (1970) ont utilisé un mode de réarrangement de la matrice permettant de placer dans le même vecteur tous les éléments inconnus (p. ex., Φ_2, Φ_3, Φ_4, q_1 et q_5 dans l'équation 3.5) et de déterminer ensuite leurs valeurs simultanément à l'aide d'une méthode directe. L'exécution de leur programme nécessite la mise en mémoire de toute la matrice carrée.

Dans le programme NETFLO, la matrice et les vecteurs de l'équation (3.5) sont réarrangés et répartis selon le degré de liberté des noeuds:

$$\begin{array}{c} \text{noeuds} \rightarrow \\ \downarrow \end{array} \begin{array}{c} 2 \\ 3 \\ 4 \\ 1 \\ 5 \end{array} \begin{bmatrix} 2 & 3 & 4 & 1 & 5 \\ (e_a+e_b+e_d) & -e_b & -e_d & -e_a & 0 \\ -e_b & (e_b+e_c) & -e_c & 0 & 0 \\ -e_d & -e_c & (e_c+e_d+e_e) & 0 & -e_e \\ -e_a & 0 & 0 & e_a & 0 \\ 0 & 0 & -e_e & 0 & e_e \end{bmatrix} \times \begin{bmatrix} \varphi_2 \\ \varphi_3 \\ \varphi_4 \\ \varphi_1 \\ \varphi_5 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ q_1 \\ q_5 \end{bmatrix} \quad (3.6)$$

L'équation matricielle peut se réduire à :

$$\begin{bmatrix} E_{ff} & E_{fc} \\ E_{cf} & E_{cc} \end{bmatrix} x \begin{bmatrix} \phi_f \\ \phi_c \end{bmatrix} = \begin{bmatrix} Q_f \\ Q_c \end{bmatrix} \tag{3.7}$$

Cette équation correspond à deux équations matricielles :

$$E_{ff}\phi_f + E_{fc}\phi_c = Q_f \tag{3.8}$$

et

$$E_{cf}\phi_f + E_{cc}\phi_c = Q_c \tag{3.9}$$

On peut utiliser l'équation (3.8) pour calculer les valeurs de charges inconnues (ϕ_f), et l'équation (3.9) permet d'obtenir le débit aux noeuds limites (Q_c). L'approche adoptée dans le programme NETFLO consiste à calculer la charge à tous les noeuds libres à l'aide de l'équation (3.8) et à calculer ensuite le débit dans chacun des segments en utilisant l'équation (3.1). L'équation (3.8) peut s'écrire sous la forme :

$$E_{ff}\phi_f = Q_f - E_{fc}\phi_c = Q_f' \tag{3.10}$$

Après avoir créé le vecteur de débit augmenté Q_f', le programme NETFLO détermine la valeur ϕ_f à l'aide de l'algorithme de Choleski (p. ex., Jenning, 1977, p. 107).

3.3 Programme NETFLO

Le programme NETFLO a été écrit dans le but d'effectuer les opérations mathématiques décrites à la section précédente. La figure 3.2 montre la structure de ce programme. En séquence, le programme NETFLO (figure 3.3) : 1) lit les conditions aux limites et les données relatives au réseau de lignes généré par le programme NETWRK, 2) définit la condition aux limites pour chaque noeud limite, 3) renumérote tous les noeuds afin de réduire la largeur de bande de la matrice E et détermine les codes pour la condensation et la mise en mémoire de la matrice, 4) détermine les valeurs de charges inconnues, 5) calcule le débit dans chaque segment, le débit total à chaque limite et les paramètres d'écoulement cumulatifs dans chaque intervalle de direction de dix degrés.

3.3.1 Opérations préliminaires

Etant donné que le programme NETFLO est appliqué à des problèmes importants sur les plans du nombre de fractures et du nombre d'intersections, les besoins de rangement en mémoire deviennent un facteur critique. Pour cette raison, on définit une dimension maximum dans les instructions de déclaration de la routine principale pour toutes les variables nécessitant un grand tableau. De plus, on assigne les mêmes valeurs maximales aux variables MAXELT, MAXNOD, MAXFRE et

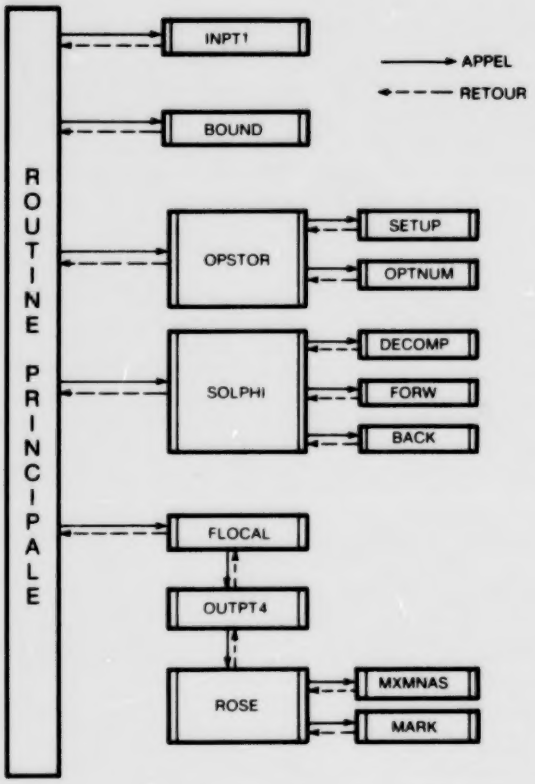


Figure 3.2. Structure du programme NETFLO.

MAXA, qui sont passées aux différents sous-programmes par une instruction **COMMON**. Ce système permet d'ajuster facilement les dimensions d'un tableau au besoin.

Le programme NETFLO lit d'abord les données relatives à la configuration des limites et aux conditions d'écoulement. Du point de vue de l'écoulement de fluide, une limite peut être qualifiée soit de limite étanche (IBC=0), soit de limite à charge fixe (IBC > 0). Le programme NETFLO lit ensuite les données relatives aux intersections (ou noeuds) générées par le programme NETWRK, c'est-à-dire les coordonnées des points d'intersection (XORD, YORD) et les numéros des limites (IB) pour toutes les intersections d'une ligne avec une limite. Enfin, il lit les données relatives aux éléments, qui sont simplement la table de connectivité (NOD1 et NOD2), et la longueur, l'orientation et l'ouverture de chaque élément.

La première étape de calcul dans le programme NETFLO consiste à assigner la valeur exacte de la charge (PHI) à chaque noeud situé sur une limite à charge fixe. Cette opération est effectuée par le sous-programme BOUND qui tient compte de trois possibilités : charge constante suivant une limite (IBC = 1), charge variant de façon linéaire (IBC = 2) ou charge variant de façon logarithmique (IBC = 3).

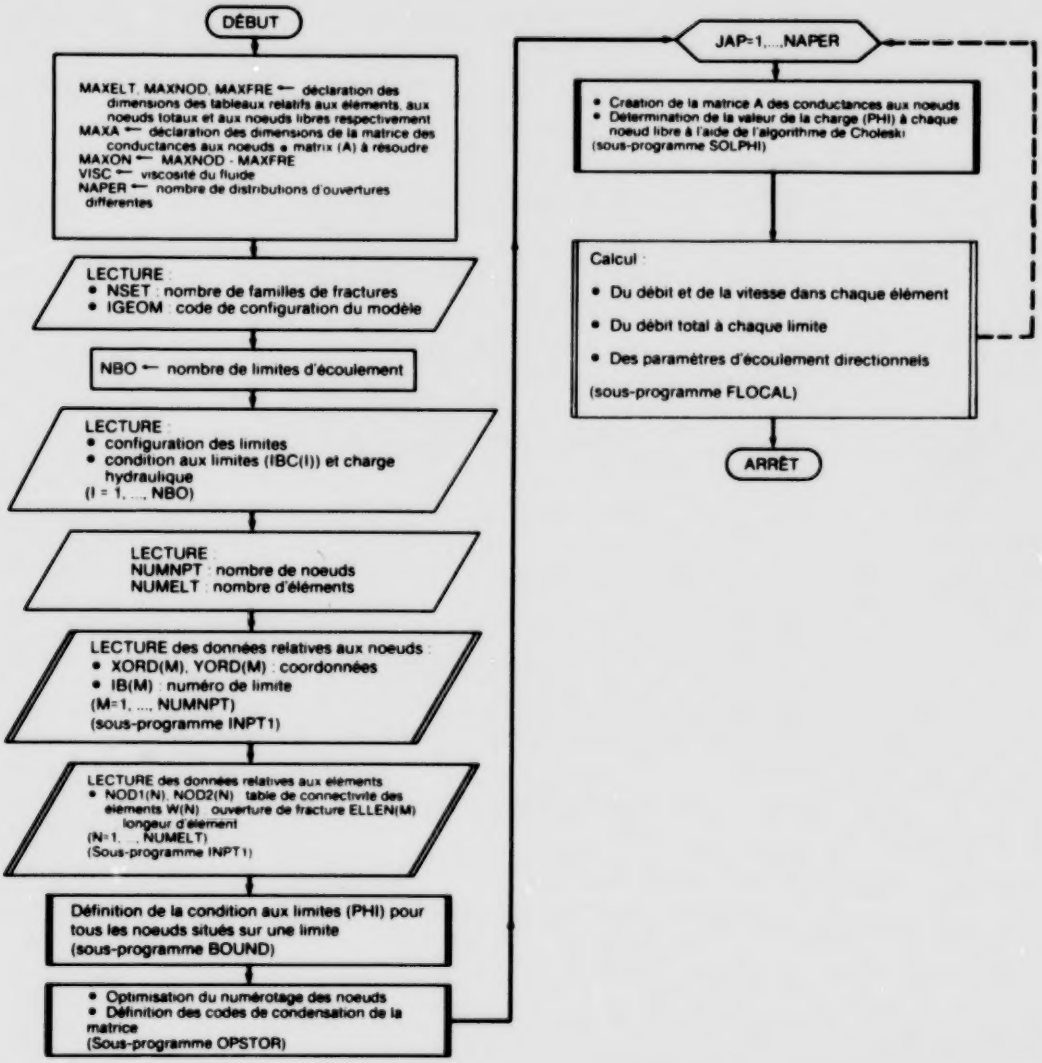


Figure 3.3. Organigramme de la routine principale du programme NETFLO.

En vue d'appliquer le programme NETFLO à des problèmes relatifs à un grand nombre de fractures, deux mesures sont prises afin d'accroître son efficacité de mémorisation : renumérotage des noeuds et stockage dans un tableau unidimensionnel de la matrice principale des coefficients d'écoulement à largeur de bande variable. Cette étape d'optimisation est effectuée principalement par le sous-programme OPSTOR (figure 3.4).

En raison du processus aléatoire utilisé pour la génération de réseaux de lignes (chapitre 2), les numéros de noeuds ont une distribution essentiellement aléatoire dans le modèle. Dans ce

numérotage initial, deux noeuds connectés peuvent avoir des numéros très espacés, ce qui peut donner une matrice A des coefficients d'écoulement très clairsemée et sans bande. Pour cette raison, le sous-programme OPSTOR renumérote systématiquement tous les noeuds du modèle, ce qui réduit considérablement la largeur de bande de la matrice A créée à une étape subséquente. Le renumérotage des noeuds est effectué par deux sous-programmes, SETUP et OPTNUM, d'après un programme écrit par Collins (1973). Des modifications ont été apportées au programme original de Collins afin qu'il puisse être appliqué aux cas de deux voies d'écoulement non connectées mais

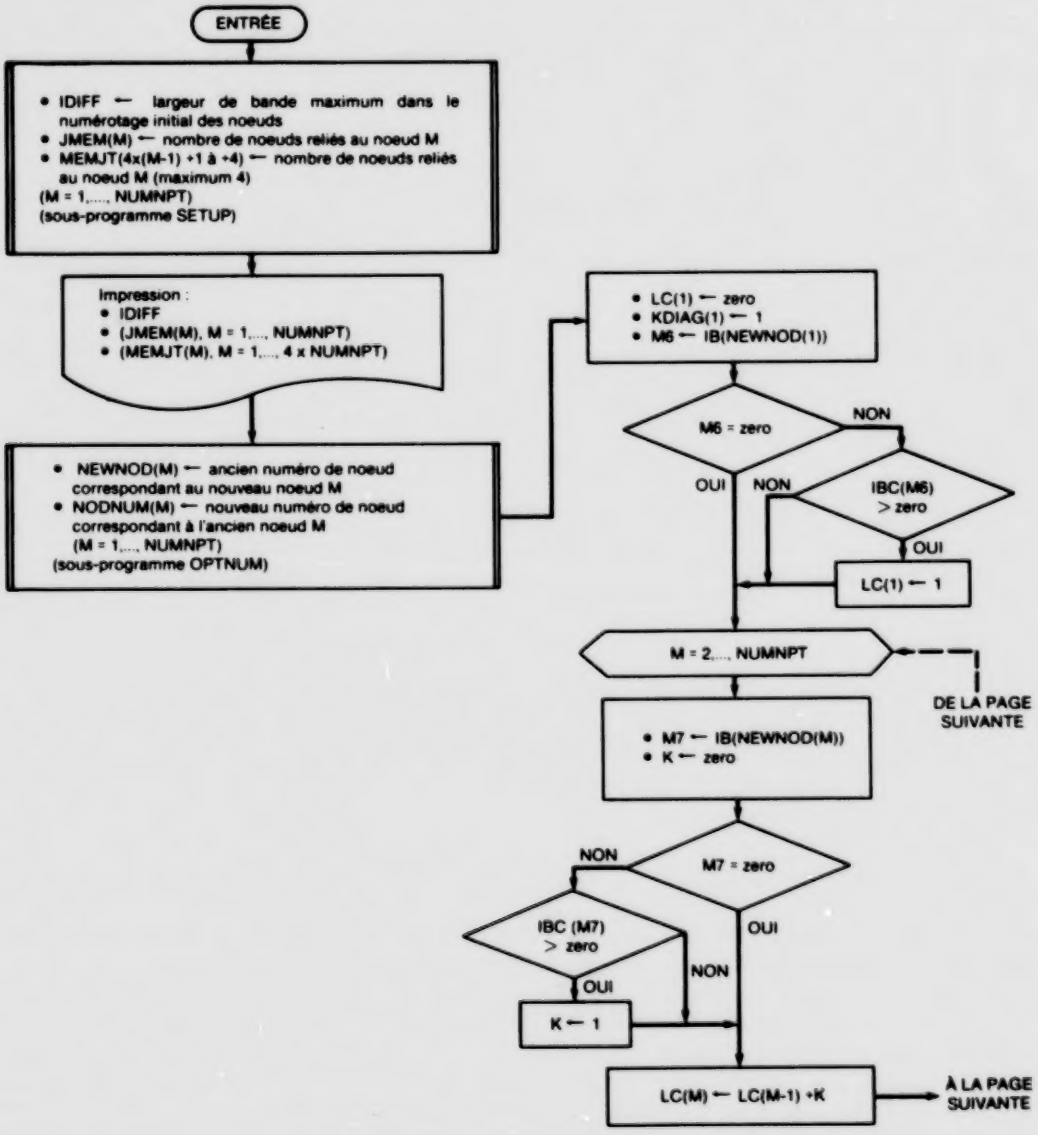


Figure 3.4. Organigramme du sous-programme OSPTOR.

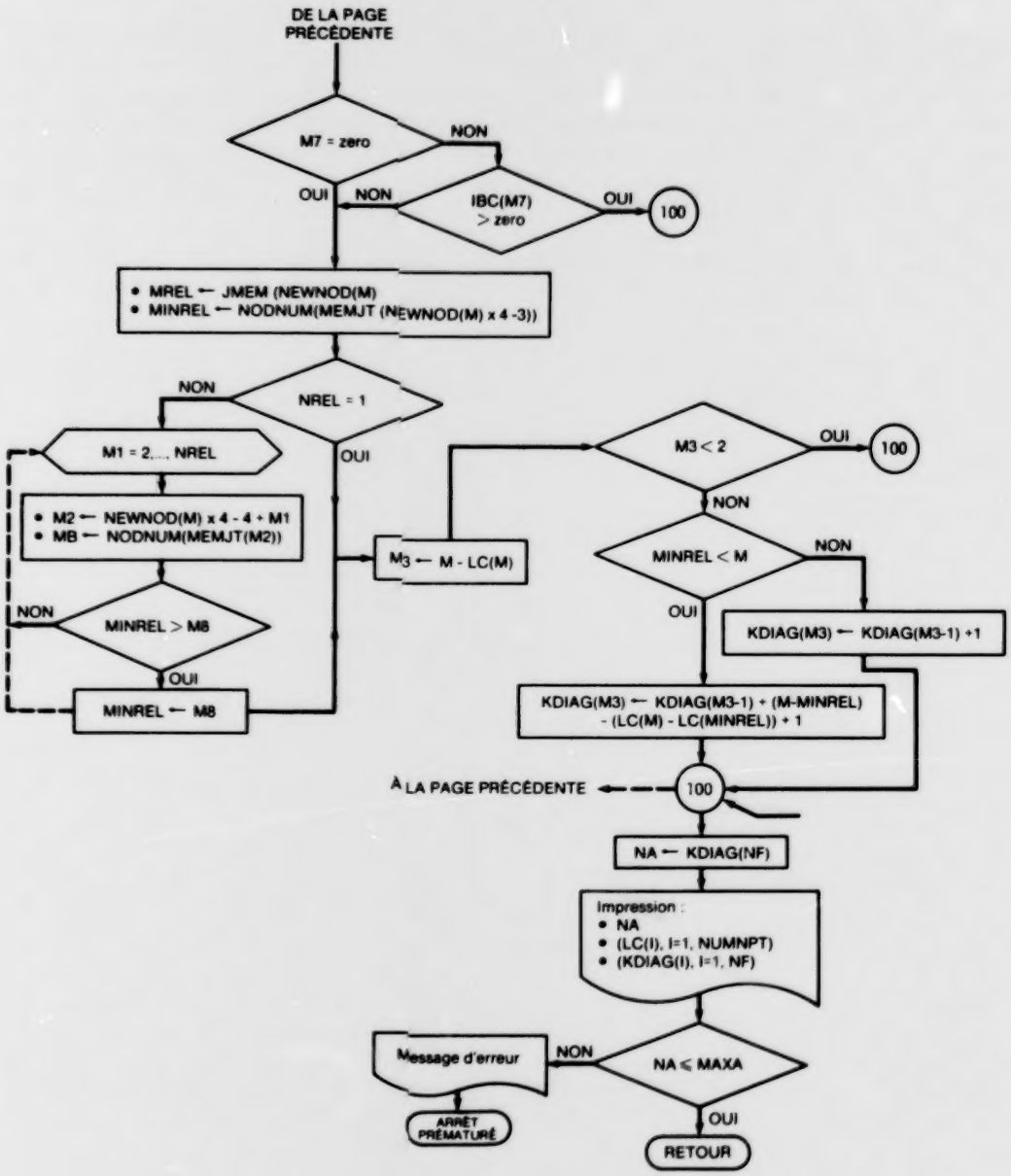


Figure 3.4. (suite).

continues ou plus. Dans ces cas, des parcours séparés et non connectés peuvent conduire le fluide d'une limite d'un modèle à l'autre.

Le sous-programme OPSTOR (figure 3.4) détermine aussi la valeur des éléments des tableaux LC et KDIAG, qui sont tous deux utilisés par la suite pour accéder aux éléments des différents tableaux utilisés dans les calculs des paramètres d'écoulement. LC est utilisé essentiellement pour la décomposition de la matrice : LC(1) prend comme

valeurs les numéros des noeuds contraints qui ont un indice inférieur ou égal à 1. Dans l'exemple de la figure 3.1, les valeurs de LC pour les numéros (indices) des noeuds correspondants sont :

$$LC(1) = LC(2) = LC(3) = LC(4) = 1 \text{ et } LC(5) = 2$$

KDIAG contient l'adresse des éléments diagonaux de la matrice A dont le calcul est fait dans une étape subséquente. A est une matrice symétrique, diagonalement dominante, correspondant à la sous-matrice E_{ff} de l'équation (3.7). Elle est mémorisée dans un tableau unidimensionnel à largeur de bande variable (Jennings, 1977, p. 97). La figure 3.5 montre une matrice symétrique 6 x 6 qui illustre la méthode de stockage à largeur de bande variable. Lorsque seulement la matrice triangulaire inférieure de la figure 3.5 est mise en mémoire, les nombres d'éléments à garder en mémoire pour les six rangées sont respectivement 1, 1, 2, 1, 5 et 4. Cette matrice peut être mise en mémoire dans le tableau unidimensionnel A :

$$\begin{matrix} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & \\ A = \{ & a_{11} & a_{22} & a_{32} & a_{33} & a_{44} & a_{51} & 0 & a_{53} & a_{54} & a_{55} & a_{63} & a_{64} & 0 & a_{66} & \} \end{matrix} \qquad (3.11)$$

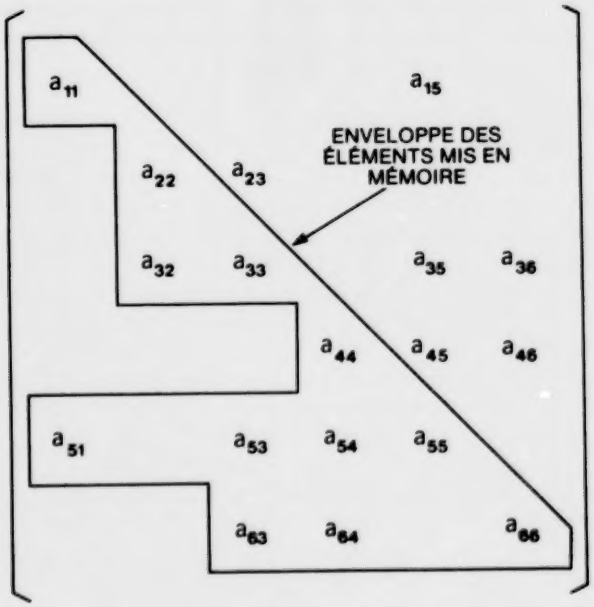


Figure 3.5. Matrice symétrique à largeur de bande variable.

En vue d'interpréter le tableau 3.11, on donne l'adresse des six éléments diagonaux dans le tableau d'entiers :

$$KDIAG = \{1\ 2\ 4\ 5\ 10\ 14\} \qquad (3.12)$$

Le stockage à largeur de bande variable est pratique pour la matrice A étant donné que même après le renumérotage des noeuds

effectué précédemment, la structure irrégulière du réseau de lignes donne de nombreuses indentations dans la matrice bande résultante. Avec un mode de stockage de bande complète, il faudrait mettre en mémoire un plus grand nombre d'éléments nuls. D'autre part, il est aussi possible d'éliminer complètement tous les éléments nuls, par exemple en utilisant deux tableaux d'entiers désignant les adresses des éléments. Cependant, le stockage complet du tableau A dont on a éliminé les zéros nécessite un programme plus complet; de plus, il est possible que cette façon de procéder ne permette pas une économie considérable d'espace de mémoire en raison du tableau d'adresses supplémentaire requis.

3.3.2 Calculs des paramètres de l'écoulement du fluide

Dans les étapes suivantes, le programme NETFLO (figure 3.3) construit les tableaux pour le calcul des paramètres de l'écoulement de fluide en utilisant le mode de stockage défini précédemment, et il détermine les valeurs des charges inconnues du système. Les deux opérations sont effectuées par le sous-programme SOLPHI (figure 3.6).

La valeur de l'expression $\gamma/12\mu$ (équation 3.1) est assignée à la constante FLUID au début du sous-programme. Ensuite, la conductance d'élément E est calculée pour chaque segment de fracture d'après l'équation (3.2) sous la forme :

$$e_N = W^3_N \times \text{FLUID}/\ell_N \qquad (3.13)$$

où N est l'indice de l'élément, W est l'ouverture de la fracture et ℓ est la longueur de l'élément. De toute évidence, chacun des deux noeuds reliés par l'élément N est soit du type libre (charge inconnue), soit du type contraint (charge fixe). Lorsqu'au moins un des noeuds est du type libre, la valeur de e est inscrite à la position appropriée dans le tableau A. Dans le cas d'un noeud contraint, la valeur de e est multipliée par la valeur de charge correspondante et le résultat est inscrit directement dans le vecteur de débit augmenté (Q_f' dans l'équation 3.10). Ce dernier vecteur est mis en mémoire dans le tableau PHI.

On résout ensuite le système matriciel pour trouver toutes les valeurs des charges inconnues à l'aide de trois sous-programmes courts (DECOMP, FORW et BACK) adaptés à partir du traitement de Jennings (1977). Le sous-programme DECOMP applique la méthode de décomposition de Choleski à la matrice A (Jennings, 1977, p. 107). Les sous-programmes FORW et BACK effectuent respectivement une substitution directe et une substitution inverse dans le système d'équations. On utilise ensuite de nouveau le tableau PHI pour mémoriser la solution de chacune de ces deux dernières opérations. Cet usage multiple du tableau PHI permet une économie de mémoire.

Après avoir terminé les calculs des charges, le sous-programme SOLPHI redonne à tous les noeuds leur numéro initial, de façon que les résultats puissent être plus faciles à interpréter. Dans une étape finale, avant de repasser la commande à la routine principale, le sous-programme SOLPHI imprime la valeur de la charge à chaque noeud.

A ce stade, la commande est passée au sous-programme FLOCAL (figure 3.7) qui calcule le débit (FR), la vitesse d'écoulement (VEL) et le nombre de Reynolds (RE) de chaque segment de fracture à l'aide des relations suivantes :

$$FR_N = e_N \times (PHI_{noeud\ 1} - PHI_{noeud\ 2}) \qquad (3.14)$$

$$VEL_N = FR_N / W_N \qquad (3.15)$$

et

$$RE_N = |FR_N / \mu| \qquad (3.16)$$

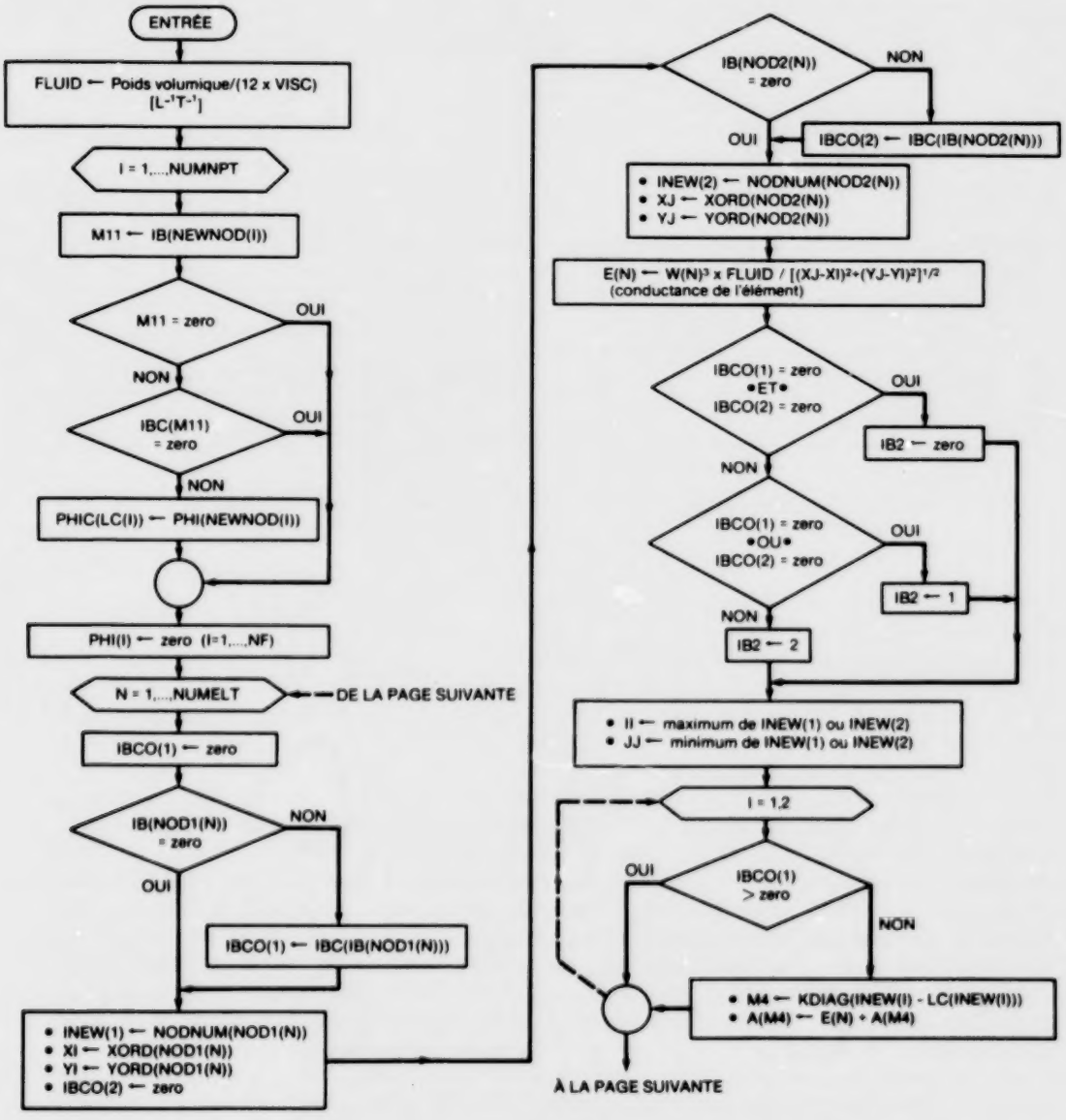


Figure 3.6. Organigramme du sous-programme SOLPHI.

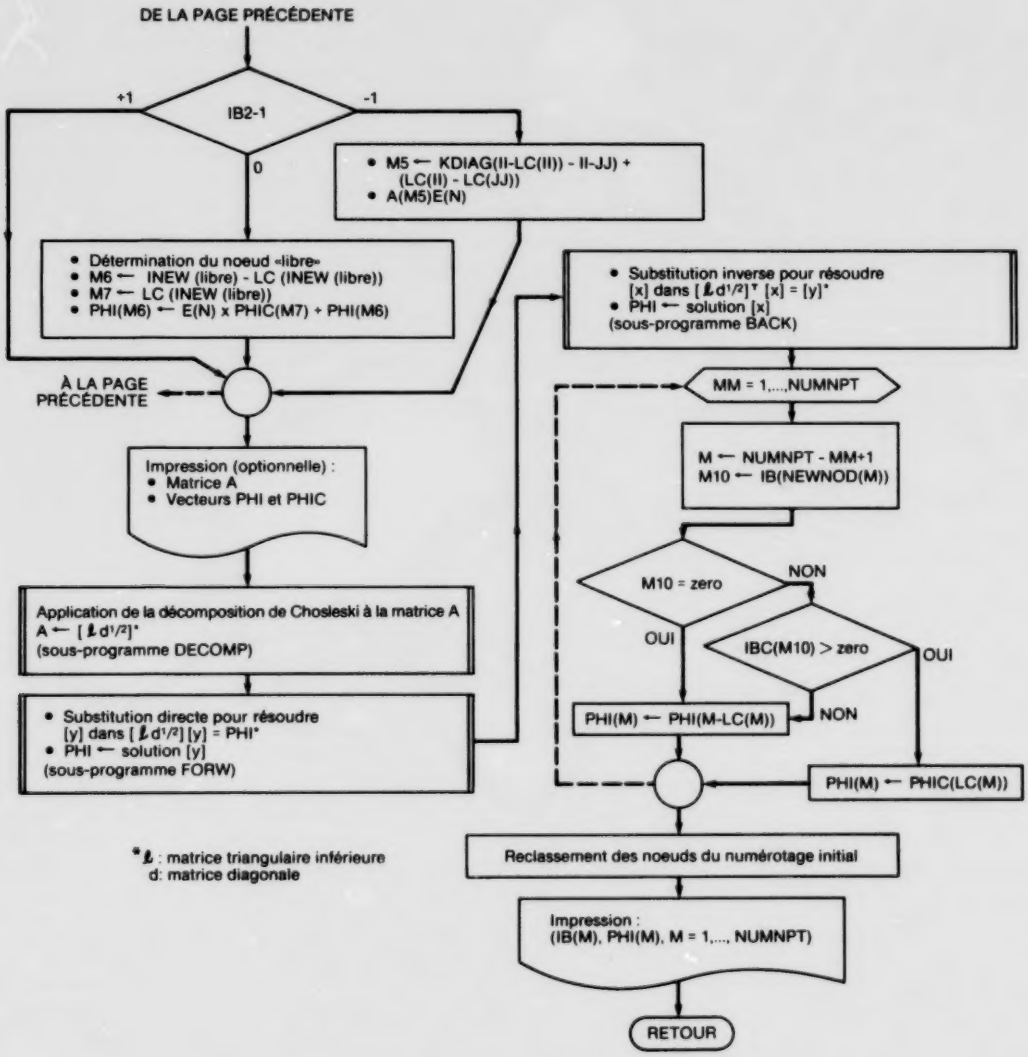


Figure 3.6. (suite).

Dans le sous-programme ELTDEF du programme NETWRK, les éléments sont définis de façon que pour tout élément le noeud numéro 1 ait la plus faible valeur de y. Avec cette convention, le débit calculé par l'équation (3.14) est positif dans le sens des y grandissants et est négatif autrement, la direction par rapport à x étant sans importance. Pendant ces calculs des débits, le débit cumulatif est calculé pour tous les éléments ayant une extrémité sur une limite. A la fin des calculs de débit, le débit total est imprimé pour chaque limite séparément.

De plus, le sous-programme FLOCAL calcule la valeur totale d'un certain nombre de variables dans chaque intervalle de direction de

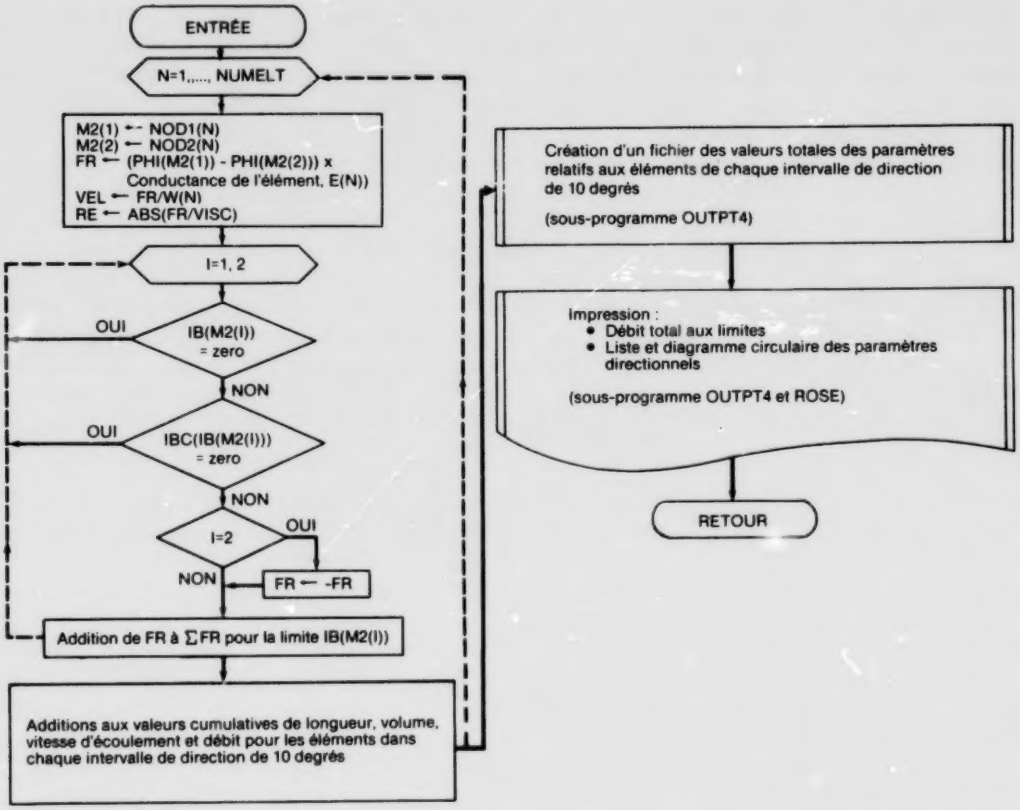


Figure 3.7. Organigramme du sous-programme FLOCAL.

10 degrés. Les principales variables directionnelles calculées dans FLOCAL sont la somme des : 1) longueurs d'éléments ℓ_N, 2) surfaces des fractures $W_N \times \ell_N$), 3) perméabilités des éléments définies par $W^2_N \times \ell_N$, 4) vitesses d'écoulement VEL_N et débits FR_N. De plus, les produits de la vitesse et d'autres paramètres sont aussi calculés pour chaque intervalle de direction de 10 degrés. Ces produits sont utilisés dans le programme NETRANS sur une base expérimentale pour calculer différentes vitesses moyennes pondérées à l'aide de différents facteurs de pondération (section 4.2.2).

3.4 Vérification du programme et évaluation de la précision

Pour vérifier les résultats du programme NETFLO, on a utilisé un modèle simple constitué de deux fractures verticales de 10 m de longueur reliées par une famille de neuf fractures horizontales espacées de 1 m (figure 3.8). On impose des charges hydrauliques de 0 et 100 m à la limite inférieure et à la limite supérieure respectivement. Le débit théorique dans chacune des fractures peut être facilement calculé à l'aide de la loi cubique du débit de fluide entre deux plaques parallèles lisses (équation 3.1). Dans ces calculs,

les valeurs choisies pour le poids volumique du fluide (γ) et la viscosité dynamique (μ) sont 9.800 x 10^3 N/m^3 et 1.002 x 10^{-3} Pa.s respectivement. En supposant une ouverture de 10 µm pour toutes les fractures, on obtient un débit dans chacune des fractures verticales de -8.1503659 x 10^{-9} m^3/s avec une précision de huit chiffres. Le signe moins signifie que l'écoulement est vers le bas.

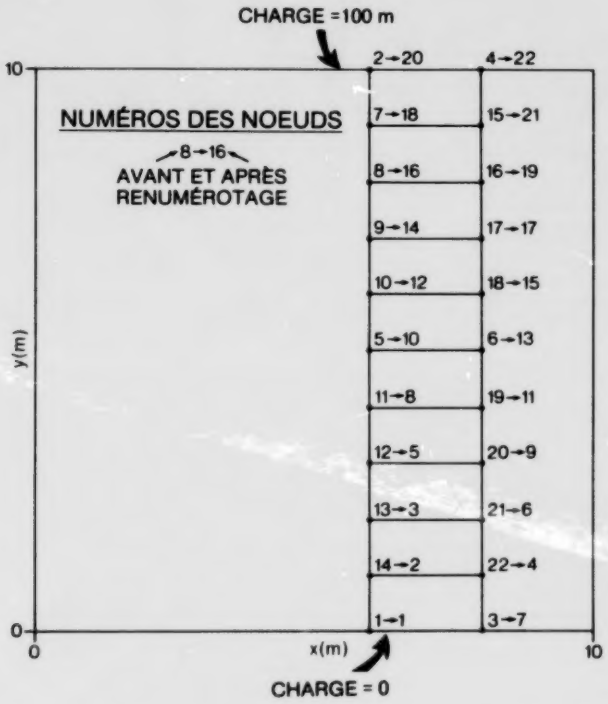


Figure 3.8. Système de fractures utilisé pour évaluer la précision du programme NETFLO.

Avant de résoudre le problème de l'écoulement de fluide représenté à la figure 3.8, le programme NETFLO doit d'abord réarranger les numéros des noeuds afin de réduire au minimum la largeur de bande de la matrice des conductances aux noeuds (équation 3.5). Les numéros de noeuds avant et après renumérotage sont donnés à la figure 3.8. Le numérotage initial a été intentionnellement effectué de façon à être inefficace; il donnerait une largeur de bande de 19 (c'est-à-dire 22-3). Le renumérotage effectué par le sous-programme OPTNUM, qui est appelé par le sous-programme OPSTOR, réduit la largeur de bande à 4, ce qui se traduit par une économie considérable de l'espace de mémoire requis.

Une valeur de débit a été calculée par le programme NETFLO pour chacun des segments de fracture compris entre deux noeuds consécutifs de la figure 3.8. Les valeurs calculées pour les segments

verticaux varient entre -8.1501810 et -8.1506499 x 10^{-9} m^3/s. Par conséquent, dans ce modèle simple, l'erreur absolue maximum sur le débit est d'environ 3 x 10^{-13} m^3/s, ce qui correspond à une erreur relative d'environ 3.5 x 10^{-5}.

Etant donné que l'ouverture est uniforme dans chaque fracture et que par conséquent l'équation de débit constitutive (équation 3.1) est linéaire, les calculs numériques devraient théoriquement donner la solution exacte. En réalité, la faible erreur numérique observée n'est pas une erreur de «discrétisation» du type le plus couramment rencontré en analyse numérique. Elle est plutôt le résultat d'un certain nombre d'erreurs d'arrondi résultant du fait que les mémoires de toutes les machines à calculer sont finies et que seulement un nombre fixe (habituellement entre 7 et 12) de chiffres peuvent être retenus après chaque opération arithmétique.

Ainsi, dans une séquence de calculs, tous les nombres irrationnels et tous les nombres comprenant plus de chiffres significatifs qu'il est possible d'en retenir doivent être arrondis. L'analyse détaillée de l'erreur d'arrondi cumulative dans les résultats finals de calculs numériques est généralement difficile étant donné que la grandeur et le signe de l'erreur de chaque opération sont imprévisibles, particulièrement dans des systèmes générés par méthode stochastique. Des modèles probabilistes peuvent être appliqués à la propagation des erreurs d'arrondi (Henrici, 1962), mais la simple approche empirique décrite dans les prochains paragraphes suffit aux fins de l'évaluation de la précision des résultats du programme NETFLO.

Afin d'évaluer l'importance de l'erreur d'arrondi cumulative produite par le programme NETFLO pour de gros systèmes, on a augmenté successivement à 20, 50 et 100 le nombre de noeuds (NN) entre les limites à charge fixe de la figure 3.8. Les erreurs résultantes sur la charge et le débit calculés sont représentées à la figure 3.9. Pour une raison inconnue, l'erreur sur la charge hydraulique est toujours positive pour NN = 10, 20 et 100, et toujours négative pour NN = 50. Il est aussi intéressant de remarquer qu'à chaque distance verticale l'erreur dans une fracture verticale est en général légèrement différente de l'erreur dans l'autre fracture verticale; par conséquent, il y a un très faible écoulement dans chacune des fractures horizontales, ce qui ne devrait pas être le cas. Cependant, ces différences sont relativement faibles et ne peuvent pas être représentées sur la figure 3.9. La plus importante conséquence de la figure 3.9 est que, contrairement à une erreur de discrétisation, une erreur d'arrondi augmente lorsque le nombre de noeuds du modèle augmente.

De toute évidence, un modèle ayant une configuration plus complexe que le système de la figure 3.8 produirait des distributions d'erreurs moins régulières que les courbes de la figure 3.9. Bien qu'elle ne soit strictement applicable qu'au modèle simple de la figure 3.8, la figure 3.9 indique néanmoins que l'erreur sur le débit est maximale près d'une limite à charge fixe puisque la pente de la courbe de l'erreur sur la charge hydraulique est plus forte en cet endroit.

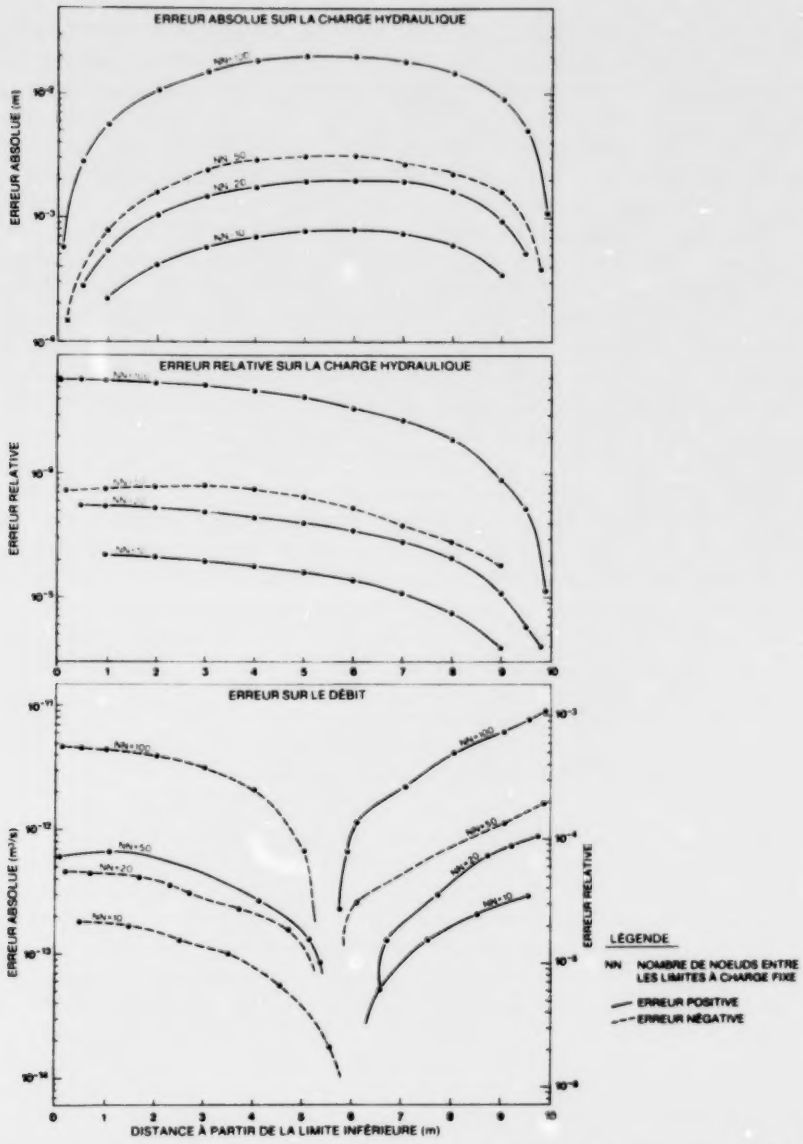


Figure 3.9. Distribution de l'erreur d'arrondi cumulative pour différents nombres de noeuds entre les limites à charge fixe du système de la figure 3.8; (A) erreur absolue sur la charge hydraulique, (B) erreur relative sur la charge hydraulique, (C) erreur sur le débit.

De plus, l'erreur sur le débit à la limite d'écoulement entrant est de signe contraire à l'erreur à la limite d'écoulement sortant. Par conséquent, une simple vérification de la différence des débits ou de l'équilibre des masses entre l'écoulement entrant et l'écoulement sortant permet une évaluation prudente de la précision des résultats pour un modèle particulier. La figure 3.10 montre qu'un accroissement

du nombre de noeuds entraîne un accroissement considérable de l'erreur sur l'équilibre des masses. Elle montre également que pour un nombre particulier de noeuds dans le système de la figure 3.8, une ouverture de fracture différente donne une erreur relative différente sur l'équilibre des masses. Cependant, cette différence d'erreur relative due à l'ouverture de fracture est relativement faible; de plus, elle n'est pas systématiquement dans le même sens (c'est-à-dire qu'un accroissement de l'ouverture peut donner soit une erreur plus grande, soit une erreur plus petite). Par une approximation grossière, consistant à extrapoler la tendance de la figure 3.10, on peut estimer que l'erreur relative sur l'équilibre des masses serait de l'ordre de 10 % pour une voie d'écoulement comprenant entre 200 et 500 noeuds entre les limites à charge fixe.

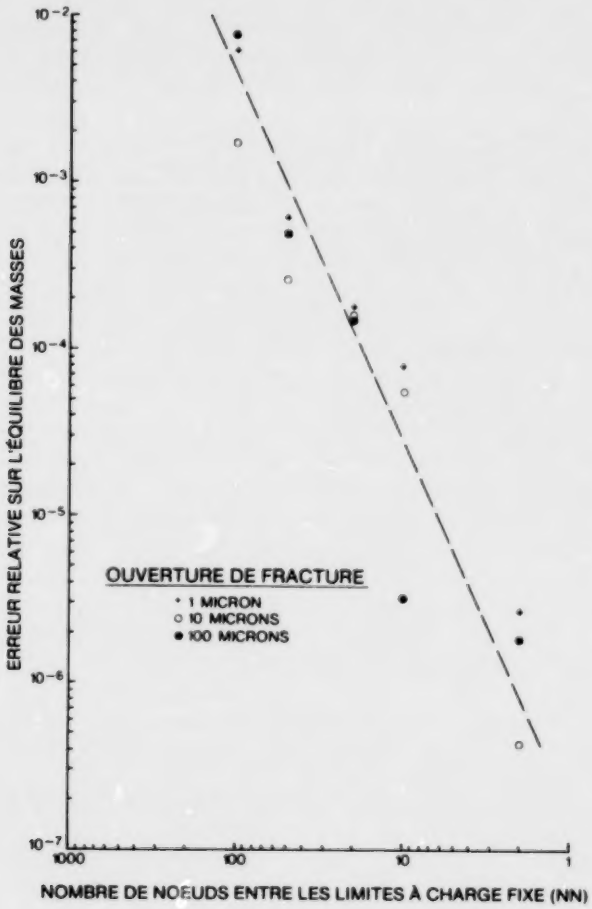


Figure 3.10. Erreur relative sur l'équilibre des masses en fonction du nombre de noeuds entre les limites à charge fixe du système de la figure 3.8.

Il y a lieu de souligner que l'erreur d'arrondi cumulative dépend à la fois du matériel informatique et du logiciel. Par exemple, l'utilisation d'une technique de résolution différente tout comme l'utilisation d'un ordinateur différent produirait des erreurs cumulatives différentes de celles obtenues dans la présente étude. Néanmoins, on peut supposer que la distribution des erreurs dans un système simple comme celui représenté à la figure 3.8 ressemblerait beaucoup à la distribution représentée à la figure 3.9, même si les valeurs réelles étaient différentes. Aucune comparaison n'a été faite dans la présente étude entre différentes techniques de résolution et différents systèmes informatiques.

Comme l'erreur d'arrondi est due à l'espace limité de mémoire réservé pour un nombre après chaque opération arithmétique, une mesure corrective évidente consisterait à accroître l'espace de mémoire réservé pour certaines variables choisies. Le programme NETFLO, dans lequel on utilise des variables en DOUBLE PRECISION pour résoudre le problème d'écoulement de fluide, donne une erreur relative sur l'équilibre des masses de l'ordre de 10^{-6} pour toutes les grosseurs de modèles représentés à la figure 3.10. Compte tenu de cette amélioration considérable, on devrait définir en DOUBLE PRECISION tous les tableaux choisis (particulièrement les tableaux A et PHI) lorsqu'on utilise un ordinateur à mots de 32 bits. Généralement, les ordinateurs à mots de 60 bits, comme ceux de la série CYBER, n'ont pas besoin de variables en DOUBLE PRECISION.

3.5 Conclusion

Le programme NETFLO a été préparé en vue d'être utilisé conjointement avec le programme NETWRK décrit au chapitre 2. Le programme est relativement rapide et simple, et il peut résoudre des problèmes relatifs à plus de 3500 éléments sans nécessiter plus de 1 méga-octet de mémoire, même lorsque certaines variables sont définies en DOUBLE PRECISION. Pour ces gros réseaux, le temps requis pour neuf calculs complets des paramètres d'écoulement (correspondant à neuf distributions d'ouvertures différentes) est de l'ordre de 120 secondes UC sur un ordinateur IBM 4341. Par conséquent, on considère que le programme NETFLO est un compromis raisonnable entre la simplicité de programmation d'une part et le rendement sur les plans du temps de calcul et de l'espace de mémoire d'autre part.

Simulation stochastique de la migration des particules dans un réseau de fractures: programme NETRANS

4.1 Introduction

On a étudié le déplacement des solutés dans des réseaux de fractures dans la roche à l'aide de modèles physiques et de modèles numériques. Krizek et coll. (1973) (aussi Castillo, 1972) ont simulé la dispersion de solutés dans un modèle numérique de fractures continues en utilisant une fonction de concentration approximative à chaque noeud. Cette fonction de concentration peut admettre soit l'absence de mélange, soit un mélange complet aux noeuds. Les hypothèses de ces chercheurs étaient basées partiellement sur les résultats d'expériences physiques sur le mélange aux noeuds et sur la dispersion dans une fracture simple (Krizek et coll., 1972; aussi Socias et coll., 1979). Wilson et Witherspoon (1976) ont aussi fait état d'expériences en laboratoire sur le mélange aux intersections des fractures.

Hull (1985) a fait état d'une autre série d'expériences physiques portant sur le mélange de solutés aux noeuds et la dispersion dans des fractures simples. Il a aussi décrit deux types de modèles physiques de réseaux de fractures servant à des études de la dispersion : 1) réseaux rectangulaires d'ouvertures continues dans une matrice imperméable de plexiglass (aussi Hull et Kuslow, 1982); 2) un modèle à double porosité d'un réseau rectangulaire d'ouvertures discontinues dans du polyéthylène poreux. Dans le cadre du même projet, Miller (1983) a élaboré un modèle de particules agissant comme traceur qui simule la dispersion de solutés dans des réseaux simples de fractures discontinues et qui tient compte de l'écoulement dans la matrice.

Schwartz et coll. (1983) ont utilisé une méthode de poursuite stochastique de particules pour simuler la dispersion de solutés dans des réseaux orthogonaux de fractures de tailles distribuées et de positions semi-aléatoires. Un modèle de transport numérique proposé par Endo et coll. (1984) tient compte du déplacement précis du fluide à l'intérieur de filets fluides dans des réseaux de fractures. Enfin, Robinson (1984) a préparé des algorithmes utilisables aussi bien dans une approche d'agglomération des masses que dans une approche de poursuite des particules.

Le modèle numérique NETRANS mentionné dans cette étude est différent des modèles susmentionnés. Il utilise un procédé

stochastique de deuxième degré basé sur les statistiques de paramètres directionnels calculés dans le programme NETFLO (chapitre 3) pour suivre les particules dans un réseau virtuel de taille arbitraire (Rouleau, 1987). Cette dernière approche a certains points communs avec l'approximation de l'hydrogramme unitaire d'un bassin versant de surface en ce sens qu'on suppose un écoulement dans des réseaux de canaux aléatoires (p. ex., Troutman et Karlinger, 1985).

4.2 Programme NETRANS

Le programme NETRANS (figure 4.1 et 4.2) utilise la valeur moyenne calculée pour chaque intervalle de direction de dix degrés des trois paramètres suivants : débit, vitesse d'écoulement et longueur de segment (ou d'élément). Le programme NETRANS permet de suivre un certain nombre de particules, une à la fois, dans un réseau de fractures virtuel. A chaque étape (ou noeud virtuel) dans la poursuite d'une particule, la direction de l'écoulement est déterminée par méthode stochastique d'après l'importance relative du débit moyen dans chaque intervalle de direction de dix degrés. Une fois que la direction de l'écoulement a été choisie, la particule est déplacée sur une distance égale à la longueur de segment moyenne dans cette direction. La vitesse de la particule est égale à une vitesse moyenne pondérée (section 4.2.2) également dans cette direction. A la fin de chaque étape, les nouvelles coordonnées des particules et le temps accru sont enregistrés. La poursuite d'une particule prend fin lorsqu'une distance donnée a été parcourue.

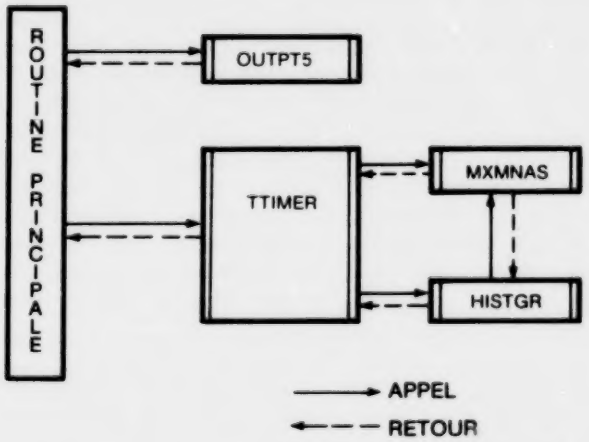


Figure 4.1. Structure du programme NETRANS.

Dans la version actuelle du programme NETRANS, la distance que parcourt une particule dans une étape donnée est simplement la moyenne de la longueur de tous les éléments pour lesquels l'écoulement est dans la direction correspondante. On pourrait perfectionner le programme en vue notamment de déterminer par méthode stochastique la distance parcourue, de la même façon qu'on détermine la direction de

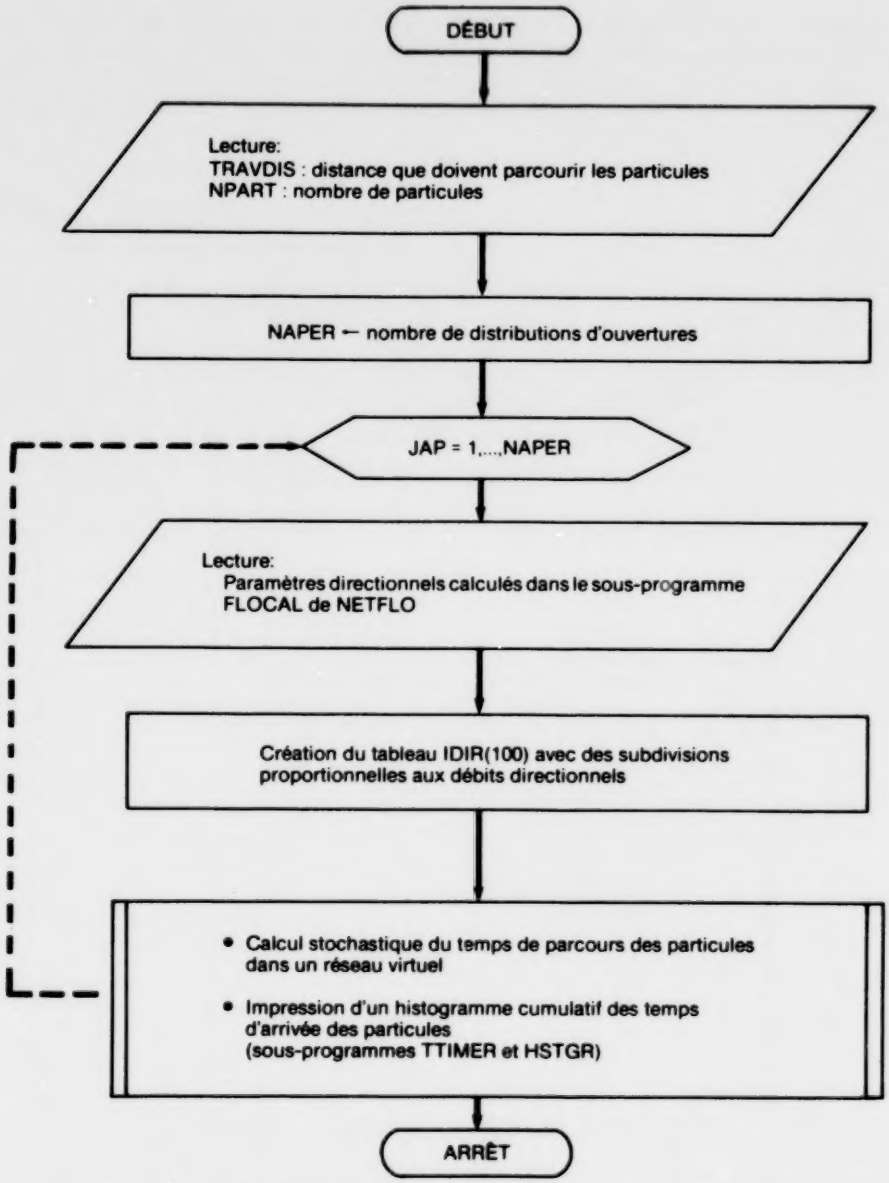


Figure 4.2. Organigramme de la routine principale du programme NETRANS.

l'écoulement. Il y a lieu d'expliquer plus en détail de quelle façon on détermine la direction de l'écoulement et la vitesse d'écoulement.

4.2.1 Détermination de la direction de l'écoulement

A la fin de l'exécution du programme NETFLO (section 3.3.2), un des paramètres directionnels enregistrés est la valeur cumulative du débit pour tous les éléments de chaque intervalle de direction de dix

degrés (DRF). Le programme **NETRANS** calcule le rapport de chacun de ces débits directionnels cumulatifs à la somme des débits dans toutes les directions. Les valeurs des rapports résultants (RDFR) sont ensuite utilisés pour subdiviser l'intervalle [0,100] : chaque subdivision

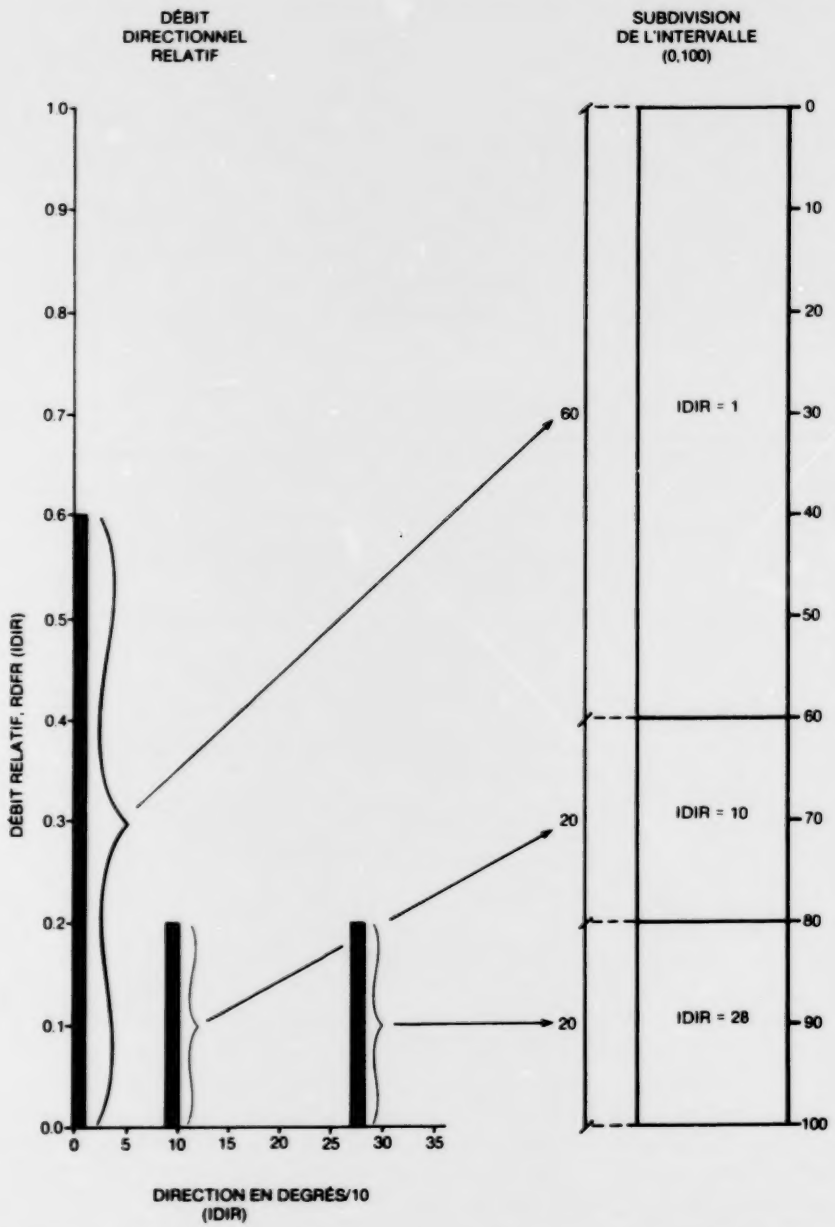


Figure 4.3. Création du tableau IDIR (100) en vue de la détermination stochastique de la direction d'écoulement.

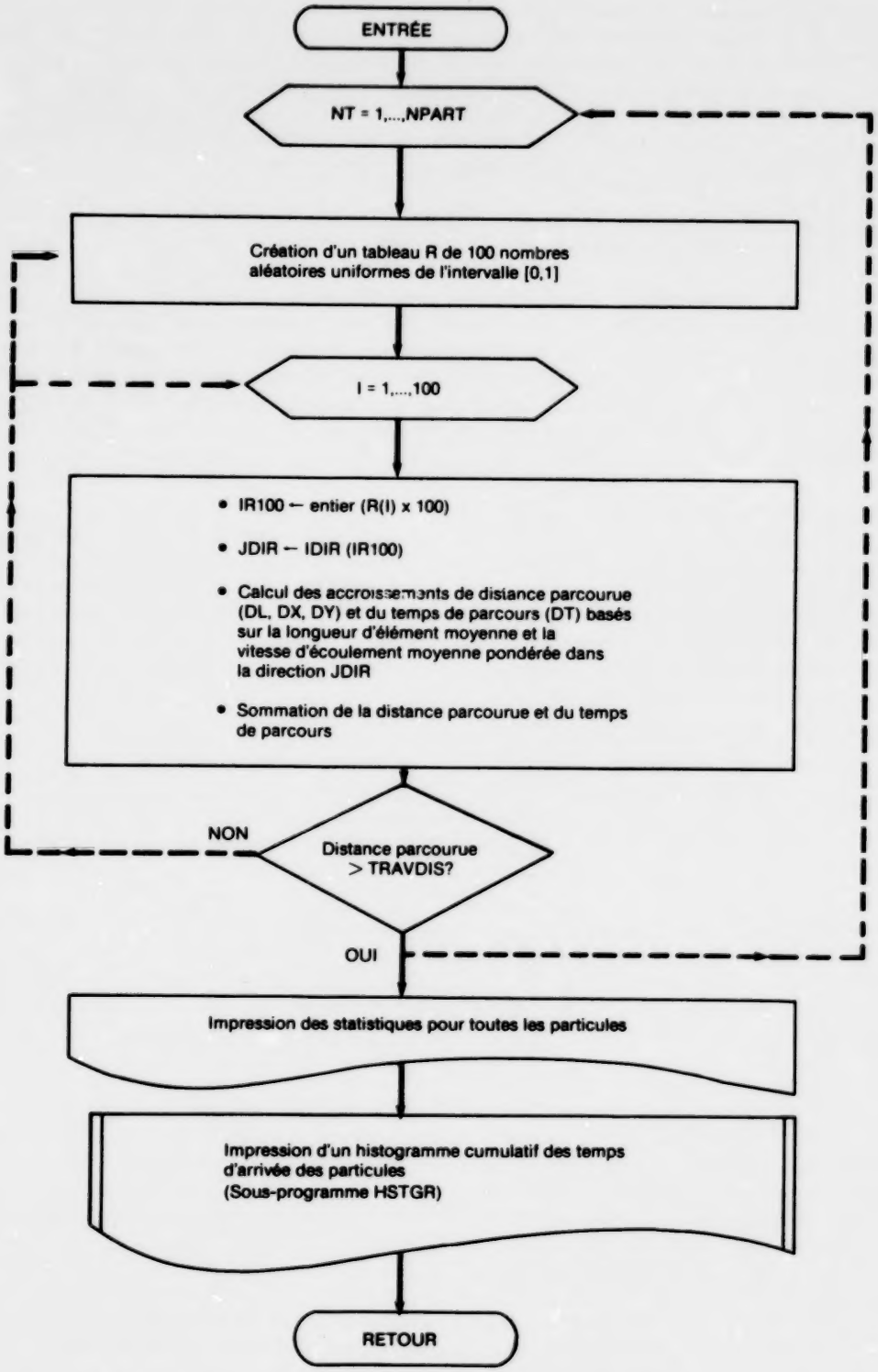


Figure 4.4. Organigramme du sous-programme TTIMER.

reçoit une valeur d'un indice de direction IDIR (entre 1 et 36); la taille de chaque subdivision est proportionnelle à la valeur de RDFR pour l'intervalle de direction correspondant (figure 4.3). Par la suite, le sous-programme TTIMER (figure 4.4) utilise le tableau IDIR, conjointement avec des nombres aléatoires distribués uniformément dans l'intervalle [0,100], pour déterminer la direction de chaque écoulement à chacune des étapes de la poursuite des particules.

4.2.2 Détermination de la vitesse d'écoulement moyenne dans chaque direction

La vitesse directionnelle moyenne simple n'est pas nécessairement la vitesse à laquelle les particules de soluté se déplacent en moyenne dans une direction donnée. Vraisemblablement, cette dernière vitesse moyenne des particules est aussi dépendante du débit dans les éléments et des longueurs des éléments. Pour cette raison, j'ai essayé un certain nombre de facteurs de pondération différents pour calculer les vitesses directionnelles moyennes pondérées. Les comparaisons des distributions résultantes des temps de parcours de particules stochastiques de l'exemple du réseau de la figure 4.5 avec les résultats déterministes des temps de parcours basés sur l'hypothèse d'un déplacement sous forme de piston sont présentées à la figure 4.6 pour quatre facteurs de pondération différents pour la vitesse directionnelle moyenne. On applique un facteur de pondération égal à l'unité en calculant la valeur cumulative de la vitesse d'écoulement simple pour tous les éléments d'un intervalle de direction

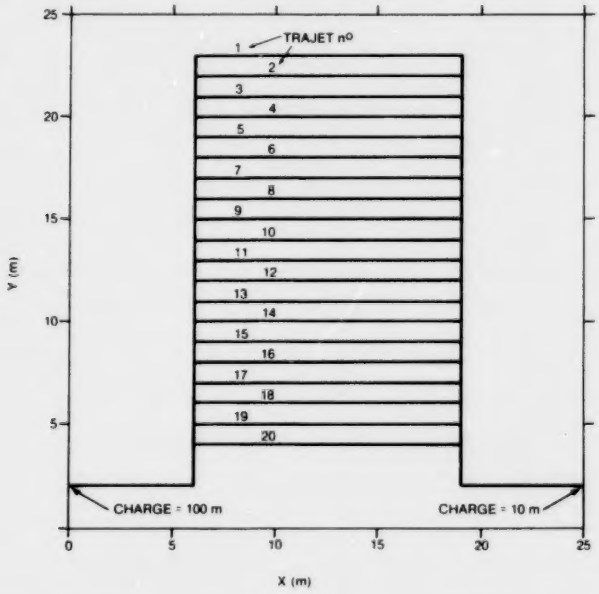


Figure 4.5. Système de fractures utilisé pour la vérification du programme NETRANS; toutes les fractures ont une ouverture de 5 µm.

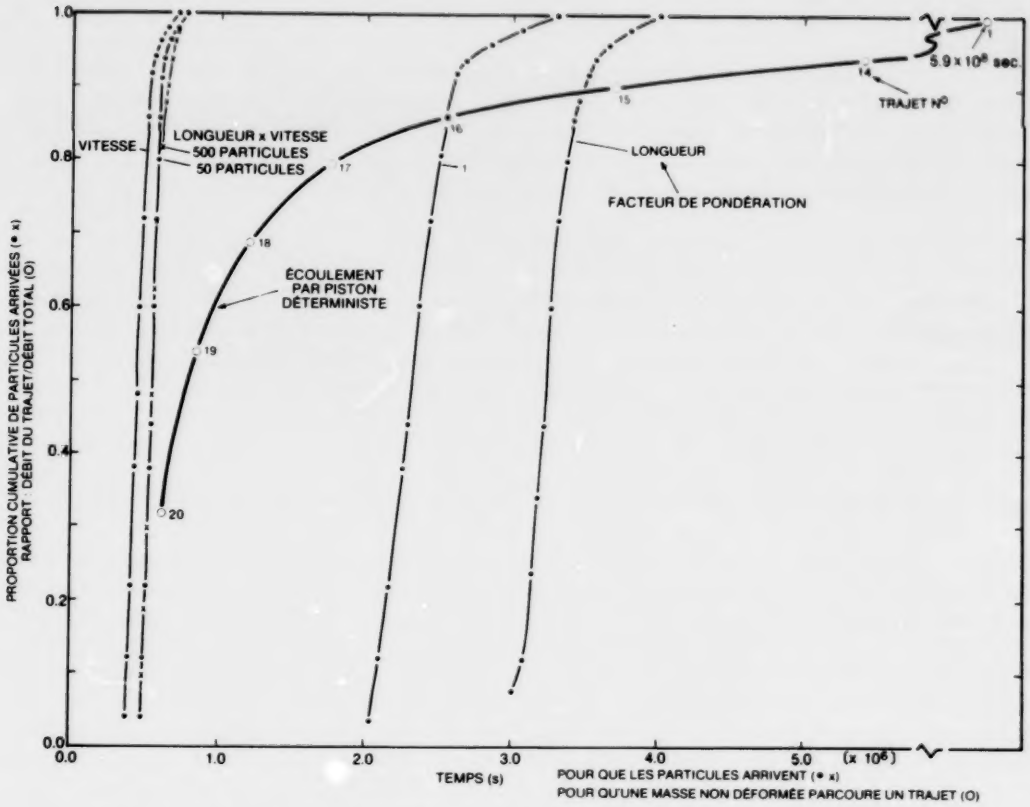


Figure 4.6. Comparaison de l'écoulement sous forme de piston déterministe et du temps de parcours stochastique de particules avec différents facteurs de pondération pour la vitesse d'écoulement.

et en divisant cette vitesse cumulative par le nombre d'éléments de l'intervalle de direction considéré. Les trois autres facteurs de pondération considérés sont la longueur de l'élément (ℓ), la vitesse d'écoulement (v) et le produit de la longueur de l'élément par la vitesse d'écoulement dans cet élément (ℓ x v).

La comparaison des temps d'arrivée de 50 % de la masse sur les courbes pour les quatre facteurs de pondération considérés (figure 4.6) montre de façon évidente que le facteur donnant les résultats les plus précis est le produit (ℓ x v). De toute évidence, lorsque tous les autres paramètres sont égaux, une vitesse d'écoulement plus élevée dans un segment de fracture signifie un débit proportionnellement plus élevé et par conséquent une plus grande probabilité qu'une particule se déplace à cette vitesse. En outre, plus un segment de fracture est long, plus les paramètres d'écoulement associés à ce segment influent sur les déplacements globaux des particules. Une analyse plus complète des facteurs de pondération possibles devrait tenir compte de l'ouverture et du débit de l'élément de même que de leur produit par la longueur de l'élément.

4.2.3 Nombre de particules requises

La figure 4.6 montre aussi les résultats d'un essai effectué avec 500 particules au lieu de 50. La courbe obtenue avec 500 particules ne diffère pas de façon appréciable de la courbe obtenue avec 50 particules, ce qui indique qu'il suffit de 50 particules pour obtenir une bonne estimation des temps de parcours.

4.3 Conclusion

Dans sa version actuelle, le programme NETRANS permet d'estimer le temps de parcours moyen de particules sur toute distance dans un réseau de fractures. La distance parcourue par les particules peut être beaucoup plus grande que la taille du réseau de fractures original généré par le programme NETWRK et utilisé dans le programme d'étude de l'écoulement de fluide, NETFLO.

Evidemment, un certain nombre d'hypothèses simplificatrices sont incluses dans le programme NETRANS et les limitations qui en résultent doivent être prises en considération dans l'interprétation des résultats. Par exemple, nous supposons une matrice imperméable et nous ne tenons pas compte de la dispersion à l'intérieur de chaque fracture. De plus, la figure 4.6 montre que la forme des courbes d'avancement des particules diffère sensiblement (beaucoup moins de dispersion) de la forme de la courbe d'avancement obtenue pour un écoulement sous forme de piston déterministe. Ce manque d'ajustement indique que la version actuelle du programme NETRANS ne permet pas d'obtenir une approximation de la forme des courbes d'avancement des solutés dans des réseaux de fractures. Les modifications du programme NETRANS qui permettraient d'obtenir des courbes d'avancement simulées reflétant mieux la réalité sont notamment le remplacement de la valeur moyenne par des valeurs distribuées, pour la longueur de segment et pour la vitesse d'écoulement pondérée dans chaque intervalle de direction.

Dans sa version actuelle, le programme NETRANS ne peut être appliqué qu'à des modèles rectangulaires. Dans la préparation d'autres sous-programmes utilisables avec des modèles circulaires, on a été confronté au problème des particules qui n'atteignent jamais la limite circulaire intérieure dans le cas d'un écoulement convergent. En effet, pour un grand nombre de particules, le trajet virtuel passait complètement à côté de la limite intérieure. Le traitement du cas de l'écoulement divergent est plus simple et on pourrait le réaliser en apportant des modifications mineures au programme, par exemple en exprimant la distance parcourue en fonction des coordonnées radiales.

Conclusion générale

Le modèle de fractures discrètes stochastiques (FDS) proposé dans le présent document est un outil pratique permettant d'intégrer les paramètres géométriques statistiques d'un système de fractures à l'étude de l'écoulement de l'eau souterraine et du transfert de masse dans des massifs rocheux fracturés. Les paramètres de la configuration des fractures qui présentent un intérêt sont les densités de fractures et les distributions des orientations, tailles et ouvertures des fractures.

On a adopté une approche modulaire pour l'écriture des programmes NETWRK, APEGEN, NETFLO et NETRANS. La séquence d'exécution de ces programmes doit correspondre à leur ordre de présentation dans le présent rapport. Cependant, chaque programme est exécuté séparément et il est possible de consulter ses résultats avant l'exécution d'un autre programme. Cette structure modulaire donne une plus grande souplesse au progiciel. De plus, les programmes ont été conçus de façon à permettre un compromis raisonnable entre l'utilisation de la mémoire, le temps de calcul et la complexité des programmes. Pour ces raisons, le progiciel de FDS convient à des simulations propres au site.

Références

Andersson, J., A.M. Shapiro, et J. Bear. 1984. A stochastic model of a fractured rock conditioned by measured information. Water Resour. Res., 20: 79-88.

Asfari, A., et P. A. Witherspoon. 1973. Numerical simulation of naturally fractured reservoirs. Soc. Pet. Eng. of AIME, Paper SPE-4290, 8 p. Dallas, Texas.

Baczynski, N.R.P. 1980. Zonal concept for spatial distribution of fractures in rock. Dans Proc. 3rd Australia-New Zealand Conf. Geomech., vol. 2, 29-33. Wellington, Nouvelle Zélande.

Baecher, G. B. 1983. Statistical analysis of rock mass fracturing, Math. Geol., 15: 329-48.

Bridges, M. C. 1975. Presentation of fracture data for rock mechanics. Dans Proc. 2nd Australia-New Zealand Conf. Geomech., 144-48. Brisbane, Australie.

Caldwell, J. A. 1972. The theoretical determination of the permeability tensor for jointed rocks. Dans Percolation through fissured rock, Proc. Symp. Int. Soc. Rock Mech. and Int. Assoc. Eng. Geol., T1C1-6. Stuttgart.

Castillo, R.E. 1972. Dispersion of contaminant in jointed rock. Thèse de Ph.D., 192 p. Northwestern University, Evanston, Ill.

Castillo, E., G. M. Karadi, et R. J. Krizek. 1972. Unconfined flow through jointed rock. Water Resour. Bull., 8(2): 266-81.

Collins, R.J. 1973. Bandwidth reduction by automatic renumbering. Int. J. Num. Methods Eng., 6: 345-56.

Conrad, F., et C. Jacquin. 1973. Représentation d'un réseau bi-dimensional de fractures par un modèle probabiliste--application au calcul des grandeurs géométriques des blocs matriciels. Rev. Inst. Fr. Pet., 28(6): 843-90.

Dershowitz, W. S., B.M. Gordon, et J. C. Kafritsas. 1985. A new three-dimensional model for flow in fractured rock. Dans Proc. Int. Congress on Hydrogeology of Rocks of Low Permeability, 441-48. Int. Assoc. of Hydrogeologists, Tucson.

Duguid, J.O. et P.C.Y. Lee. 1977. Flow in fractured porous media. Water Resour. Res., 13: 558-66.

Endo, H.K., J.C.S. Long, C.R. Wilson, et P. A. Witherspoon. 1984. A model for investigating mechanical transport in fracture networks. Water Resour. Res., 20(10): 1390-1400.

Fawcett, R.J., S. Hibbert, et R.N. Singh. 1984. An appraisal of mathematical models to predict water inflows into underground coal workings. Int. J. of Mine Water, 3: 33-54.

Gale, J.E. 1977. Etude numérique, sur le terrain et au laboratoire de l'écoulement des fluides dans un système de fractures déformables. Etude n° 72, Série scientifique, 145 p. Direction des ressources en eau, Direction générale des eaux intérieures, Environnement Canada (Aussi, Ph.D. diss., University of California, 1975.)

Gale, J.E., et A. Rouleau. 1984. Characterizing and interpreting the geometry, permeability and porosity of fractures for repository evaluation. Dans Proc. Int. Symp. Field Measurements in Geomech., Zurich, préparé par K. Kovari, 1343-69. Rotterdam: A.A. Balkema.

Henrici, P. 1962. *Discrete Variable Methods in Ordinary Differential Equations*, 407 p. New York: John Wiley.

Hudson, J.A., et P.R. LaPointe. 1980. Printed circuits for studying rock mass permeability. Technical Note, Int. J. Rock Mech. Min. Sci., 17: 297-301.

Hudson, J.A., et S.D. Priest. 1979. Discontinuities and rock mass geometry. Int. J. Rock Mech. Min. Sci., 16: 239-362.

Huitt, J.L. 1956. Fluid flow in simulated fractures. AICHE J. 2(2): 259-64.

Hull, L.C. 1985. Physical model studies of dispersion in fracture systems. Idaho Nat. Eng. Lab. Informal Report, EGG-ELS-6845, 74 p. Idaho Falls, Idaho.

Hull, L.C., et K.N. Koslow. 1982. Dispersion in fracture networks. Dans Proc. 8th Stanford Geothermal Reservoir Eng. Conf., 258-88, Stanford University, Stanford, Calif.

Huskey, W.L., et P.B. Crawford. 1967. Performance of petroleum reservoirs containing vertical fractures in the matrix. Soc. Pet. Eng. J., juin 1967: 221-28.

Isaacs, L.T., et K.G. Mills. 1980. Linear theory methods for pipe networks analysis. J. Hydraul. Div. Proc. Am. Soc. Civ. Eng., 106(HY7): 1191-1201.

Jennings, A. 1977. *Matrix Computation for Engineers and Scientists*, 330 p. New York: John Wiley.

Kesavan, H.K., et M. Chandrashekar. 1972. Graph-theoritic model for pipe network analysis. J. Hydraul. Div. Proc. Am. Soc. Civ. Eng., 98(HY2): 345-64.

Krizek, R.J., E. Castillo, et G.M. Karadi. 1973. Theoretical study of dispersion in a fractured rock aquifer. J. of Geophys. Res., 78(3): 558-73.

Krizek, R.J., G.M. Karadi, et E. Socias. 1972. Dispersion of a contaminant in fissured rock. Dans Percolation through fissured rock, Proc. Symp. Int. Soc. Rock Mech. and Int. Assoc. Eng. Geol., T3C1-15. Stuttgart.

LaPointe, P.R., et J.A. Hudson. 1985. Characterization and interpretation of rock mass jointing patterns. Geol. Soc. Am., Spec. Pap., 199, 37 p. Boulder, Colorado.

Long, J.C.S., P. Gilmour, et P.A. Witherspoon. 1985. A model for steady fluid flow in random three-dimensional networks of disc-shaped fractures. Water Resour. Res., 21: 1105-15.

Long, J.C.S., J.S. Remer, C.R. Wilson, et P.A. Witherspoon. 1982. Porous media equivalents for networks of discontinuous fractures. Water Resour. Res., 18(3): 645-58.

Louis, C. 1969. A study of groundwater flow in a jointed rock and its influence on the stability of rock masses. Imperial College Rock Mech. Res. Rep. No. 10, 90 p. Londres, U.K.

Marcus, H., et D.E. Evenson. 1961. Directional permeability in anisotropic porous media. University of California, Berkely, Water Resources Center Contrib. 31, 105 p.

Mah, R.S.H., et M. Shacham. 1978. Pipe network design and synthesis. In *Advances in Chemical Engineering*, préparé par T.B. Drew et coll., 125-209. New York: Academic Press.

Miller, J.D. 1983. A fundamental approach to the simulation of flow and dispersion in fractured media. Dans Proc. 9th Stanford Geothermal Reservoir Eng. Conf., 373-79. Stanford University, Stanford, Calif.

Noorishad, J., M.S. Ayatollahi, et P.A. Witherspoon. 1982. A finite-element method for coupled stress and fluid flow analysis in fractured rock masses. Int. J. Rock Mech. Min. Sci., 19: 185-193.

Ollos, G. 1963. Examen hydraulique de l'écoulement dans des roches crevassées sur des modèles réduits. Bull. Int. Assoc. Sci. Hydrol., 8(2): 9-22.

Parsons, R.W. 1966. Permeability of idealized fractured rock. Soc. Pet. Eng. J., juin 1966: 126-36.

Prats, M. 1972. The influence of oriented arrays of thin impermeable shale lenses or of highly conductive natural fractures on apparent permeability anisotropy. J. Pet. Technol., Oct. 1972: 1219-21.

Robinson, P.C. 1982a. Connectivity of fracture systems: a percolation theory approach. Theor. Phys. Div., At. Energy Res. Establ., Rep. TP 918, 35 p. Harwell, U.K.

Robinson, P.C. 1982b. NAMNET: network flow program. At. Energy Res. Establ., Rep. R-10510, 28 p. Harwell, U.K.

Robinson, P.C. 1984. Connectivity, flow and transport in network models of fractured media. Thèse de Ph.D., Oxford University. (Aussi At. Energy Res. Establ., Rep. TP 1072, Harwell, U.K.)

Rouleau, A. 1987. A stochastic particle transport model based on directional statistics of flow through fracture networks. Actes du NATO Advanced Research Workshop on Advances in Analytical and Numerical Groundwater Flow and Quality Modelling. Lisboa: D. Reidel Publ.

Rouleau, A., et J.E. Gale. 1985. Statistical characterization of the fracture system in the Stripa granite, Suède. Int. J. Rock Mech. Min. Sci., 22: 353-67.

Rouleau, A., et J.E. Gale. 1987. Stochastic discrete fracture simulation of groundwater flow into an underground excavation in granite. Int. J. Rock Mech. Min. Sci., 24: 99-112

Samaniego, A., et S.D. Priest. 1984. The prediction of water flows through discontinuity networks into underground excavations. Dans Proc. Symp. Design and Performance of Underground Excavations, E.T. Brown et J.A. Hudson, 157-64. Int. Soc. for Roch Mech., Cambridge.

Schwartz, F.W., L. Smith, et A.S. Crowe. 1983. A stochastic analysis of macroscopic dispersion in fractured media. Water Resour. Res., 19(5): 1253-65.

Shapiro, A.M., et J. Anderson. 1985. Simulation of steady-state flow in three-dimensional fracture networks using the boundary-element method. Adv. Water Resour. 8: 106-110.

Smith, L., et F.W. Schwartz. 1984. An analysis of the influence of fracture geometry on mass transport in fractured media. Water Resour. Res., 20: 1241-52.

Snow, D.T. 1969. Anisotropic permeability of fractured media. Water Resour. Res., 5: 1273-89.

Socias, E.J., R.J. Krizek, et R.H. Borden. 1979. Model study of hydrodynamic dispersion in jointed rock. Dans Proc. World Congress on Water Resources, 2967-73. Mexico, Mexique.

Troutman, B.M., et M.R. Karlinger. 1985. Unit hydrograph approximations assuming linear flow through topologically random channel networks. Water Resour. Res., 21(5): 743-54.

Veneziano, D. 1978. Probabilistic model of joints in rock. Res. Rep. Dept. Civ. Eng., 46 p. Massachusetts Institute of Technology, Cambridge, Mass.

Wilson, C.R., et P.A. Witherspoon. 1970. An investigation of laminar flow in fractured porous rocks. Internal rep., Dept. Civ. Eng., 178 p. University of California, Berkeley.

Wilson, C.R., et P.A. Witherspoon. 1976. Flow interference effects at fracture intersections. Water Resour. Res., 12(1): 102-104.

Wittke, W. 1970. Three-dimensional percolation of fissured rock. In Planning open pit mines, Proc. Symp. Inst. Mine Met., préparé par P.W.J. van Rensburg, 191-91. Johannesburg, Afrique du Sud.

Annexe A

Liste des principales variables utilisées dans les programmes NETWRK (NW), APEGEN (AP), NETFLO (NF), et NETRANS (NT)

Liste des principales variables utilisées dans les programmes NETWRK (NW), APEGEN (AP), NETFLO (NF), et NETRANS (NT)

A.1 Liste des paramètres d'entrée

AAP(I) et BAP(I) (NW,AP) : premier et deuxième paramètres de la distribution des ouvertures de la famille de fractures I. Lorsque APDSTR=1, AAP est la valeur de l'ouverture et BAP n'est pas utilisé. Lorsque APDSTR=2, AAP et BAP sont respectivement la moyenne et l'écart type de la valeur de l'ouverture remplacée par un logarithme (naturel). Remarque : l'ouverture est exprimée en mètres; (m).

ALE(I) et BLE(I) (NW) : premier et deuxième paramètres de la distribution des longueurs de traces de la famille de fractures I. Lorsque LDSTR=1, ALE est la longueur de trace et BLE n'est pas utilisé. lorsque LDSTR=2, ALE et BLE sont respectivement la moyenne et l'écart type des valeurs de longueurs de traces remplacées par des logarithmes (naturels); (m).

APDSTR(I) (NW, AP) : type de distribution des ouvertures de la famille de fractures I : (1) variable à une seule valeur, (2) distribution log-normale; voir AAP et BAP.

ATH(I) et BTH(I) (NW) : premier et deuxième paramètres de la distribution des angles de traces à l'intérieur du plan de simulation de la famille de fractures I. Lorsque THDSTR=1, ATH est la valeur de l'angle et BTH n'est pas utilisé. Lorsque THDSTR=2, ATH et BTH sont respectivement la moyenne et l'écart type des valeurs des angles. ATH(I) doit être à l'intérieur de l'intervalle 0°-180°. Les valeurs des angles sont évaluées dans le sens anti-horaire à partir de l'axe des x positifs; voir figure D.1; (degrés).

ATHP(I) et BTHP(I) (NW) : premier et deuxième paramètres de la distribution des angles entre le plan de simulation et les fractures de la famille I; ATHP(I) doit être à l'intérieur de l'intervalle 0°-180°. Les angles sont exprimés comme ils sont mesurés à l'intérieur de la tranche de roche simulée, du côté droit de la trace de fracture lorsque l'observateur regarde dans la direction ATH(I); voir figure D.1; (degrés). Lorsque THDSTR=1; ATHP est la valeur de l'angle et BTHP n'est pas utilisé. Lorsque THDSTR=2, ATHP et BTHP sont respectivement la moyenne et l'écart type des valeurs des angles.

BAP(I) (NW, AP) : voir AAP(I).

BLE(I) (NW) : voir ALE(I).

BTH(I) (NW) : voir ATH(I).

BTHP(I) (NW) : voir ATHP(I).

COEF (NW) : coefficient utilisé pour multiplier un nombre estimé préliminaire de fractures d'une famille en vue d'obtenir une valeur de ESTNFR (voir sous-programme GENLIN) assez grande pour la plupart des réalisations; généralement, COEF doit être dans l'intervalle 2.0 à 6.0.

DAAP (NW, AP) : accroissement entre des valeurs différentes du paramètre de distribution des ouvertures AAP.

DENS(I) (NW) : densité de la famille de fractures I, définie comme la surface de fractures par unité de volume de roche; (m^{-1}).

HDB(I) et HDE(I) (NF) : valeurs d'entrée de la charge hydraulique au début et à la fin respectivement du segment limite I, déterminées en se déplaçant dans le sens horaire suivant la limite; n'est pas utilisé pour des limites étanches; voir figure D.3; (m).

IBC(I) (NW, NF) : code de condition aux limites pour l'écoulement suivant le segment limite I; (0) limite étanche, (1) charge constante suivant la limite, (2) charge décroissant de façon linéaire suivant la limite, (3) charge décroissant de façon logarithmique

suivant la limite; dans ce dernier cas, la limite ne doit pas traverser un axe des zéros; voir figure D.3.

IGEOM (NW,NF,NT) : code pour la configuration générale du modèle; (1) pour un modèle rectangulaire, (2) pour un modèle circulaire; voir figure D.3.

IOR (AP) : code pour le mode de génération des ouvertures; (0) lorsqu'on n'utilise pas les données d'orientation, (1) lorsqu'on tient compte de l'orientation; la deuxième option n'est pas encore utilisable.

IPRT (NW,AP,NF,NT) : code pour le degré d'informations voulu dans les fichiers de sortie; les valeurs possibles sont 1, 2 et 3, par ordre croissant du degré d'informations (voir tableau B.2). De plus, les numéros de fractures et de noeuds ne sont pas écrits sur le graphique produit par ordinateur lorsque IPRT = 1.

ISHAP(I) (NW,NF) : code pour la forme du segment limite I; (0) pour un segment de longueur nulle, (1) pour un segment linéaire, (2) pour un segment circulaire; voir figure D.3.

ISP (NW) : code pour le type d'échantillonnage des valeurs d'espacement des traces des fractures d'une même famille; (0) lorsqu'on ne désire aucun échantillonnage, (1) lorsqu'on désire un échantillonnage suivant des lignes perpendiculaires à l'orientation moyenne des traces (sous-programme SPCNG1), (2) lorsqu'on désire un échantillonnage suivant des lignes orientées de façon aléatoire (sous-programme SPCNG2).

ITERM (I) (NW) : type d'extrémités pour la famille de fractures I; (0) extrémités libres, (1) l'extrémité un (c'est-à-dire plus grande valeur de y) aboutit sur les fractures d'une famille quelconque dont l'indice <I; ITERM(1) ne peut pas être égal à 1.

LDSTR(I) NW) : type de distribution des longueurs de traces de la famille de fractures I; (1) pour une variable à une seule valeur, (2) pour une distribution log-normale; voir ALE et BLE.

MPART(10) (NW) :	tableau des nombres de particules pour lesquels on désire un enregistrement détaillé de la poursuite; il existe **NMAP** particules de ce type.
NAAP (NW,AP,NF,NT) :	nombre de distributions d'ouvertures à générer avec les mêmes paramètres de distribution.
NMAP (NT) :	nombre de particules pour lesquelles on désire un enregistrement détaillé de la poursuite (voir **MPART**).
NPART (NT) :	nombre de particules à poursuivre (généralement 50).
NSET (NW,AP,NF) :	nombre de familles de fractures; maximum = 5.
PLTSIZ (NW) :	longueur (en pouces) du plus long axe du graphique à générer; lorsque PLTSIZ $\leq$ 0, il n'y a pas de traçage de graphique.
R(I) (NW) :	rayon du segment limite I; variable utilisée seulement pour un segment circulaire, c'est-à-dire lorsque ISHAP(I) = 2; voir figure D.3; (m).
SEED (NW,AP) :	valeur de départ des sous-programmes de génération des nombres aléatoires; nombre à double précision de l'intervalle 1 à 2147483647.
THDSTR(I) (NW) :	type de distribution des angles pour la famille de fractures I; (1) pour une variable à une seule valeur, (2) pour une distribution normale; voir **ATH**, **BTH**, **ATHP** et **BTHP**.
TRAVDIS (NT) :	distance que doit parcourir une particule pour traverser le système (m); sa valeur n'a pas besoin d'être égale à la taille du modèle pour le réseau généré.
XB(I) et YB(I) (NW,NF) :	coordonnées du début du segment limite I, déterminées en se déplaçant dans le sens horaire suivant les limites; (m).
XMAXI, XMINI, YMAXI YMINI (NW) :	limites intérieures du modèle rectangulaire; voir figure D.3; (m).
XMAXO, XMINO, YMAXO et YMINO (NW) :	limites extérieures du modèle rectangulaire; requises même pour un modèle à

	configuration circulaire; voir figure D.3; (m).
YB(I) (NW,NF) :	voir XB(I).
YMAXI et YMNI (NW) :	voir XMAXI.
YMAXO et YMINO (NW) :	voir XMAXO.

A.2 Liste des paramètres d'utilisation interne

A(MAXA) (NF) :	sous-matrice des conductances aux noeuds; elle ne contient des données que pour les éléments dont les deux extrémités sont libres; A est symétrique et utilise un mode de stockage à largeur de bande variable (voir KDIAG).
ACTLEN (NW) :	longueur totale réelle de la famille présentement générée à la fin du processus de génération; (m).
AP(J) (NW) :	ouverture de la fracture J de la famille présentement générée; (m).
CUMX et CUMY (NT) :	distances cumulatives parcourues dans la direction des x et des y respectivement par la particule présentement suivie; (m).
CUMT (NT) :	temps de déplacement cumulatif pour la particule présentement suivie; (s).
CUT1(J,I) et CUT2(J,I); (NW) :	paramètre indiquant si les extrémités 1 et 2 respectivement de la ligne (J,I) tombent ou non sur une limite et indiquant aussi le type de limite : (0) aucune intersection avec une limite, (1) intersection avec une limite étanche, (2) intersection avec une limite à charge fixe.
DEADLE(I) (NW) :	longueur de segments sans issue dans la famille de fractures I; (m).
DEADVO(I) (NW) :	«volume» total (c'est-à-dire longueur x ouverture x 1) de segments sans issue dans la famille de fractures I; (m^3).
DEGRE(J) (NW) :	angle de trace à l'intérieur du plan de simulation de la fracture J de la famille présentement traitée; même convention que pour ATH(I); (degrés).

DEGREP(J) (NW) : angle entre le plan de simulation et la fracture J de la famille présentement traitée; même convention que pour ATHP(I); (degrés).

DESLEN (NW) : longueur totale de traces de fractures désirée pour la famille présentement traitée; (m).

DFR(K) (NF,NT) : somme des débits dans les éléments à l'intérieur de l'intervalle de direction K, avec K = 1 à 36; (m^3/s).

DK(K) (NF) : somme des «perméabilités» des éléments (non orientés) dans la direction K, avec K = 1 à 18; la perméabilité d'un élément est définie comme l'ouverture au carré pondérée par la longueur de l'élément, c'est-à-dire W^2 x ELLEN; (m^3).

DLE(K) (NF, NT) : somme des longueurs d'éléments (orientés) (ELLEN) dans l'intervalle de direction K, avec K = 1 à 36; (m).

DLE180(K) (NF) : somme des longueurs d'éléments (non orientés) (ELLEN) dans l'intervalle de direction K, avec K = 1 à 18; (m).

DLM(K) (NT) : longueur moyenne des éléments (orientés) dans l'intervalle de direction K, avec K = 1 à 36; (m).

DLVEL(K) (NF,NT) : somme des produits ELLEL x VEL pour tous les éléments de l'intervalle de direction K, avec K = 1 à 36; (m^2/s). DLVEL est utilisé comme la somme des facteurs de pondération dans le calcul de la vitesse directionnelle moyenne pondérée DWTVM(K).

DLVEL2(K) (NF,NT) : somme des vitesses d'écoulement pondérées dans les éléments à l'intérieur de l'intervalle de direction K, avec K = 1 à 36. La vitesse d'écoulement pondérée dans un élément est définie comme le produit (ELLEN x VEL) x VEL; (m^3/s^2).

DPOR(K) (NF) : somme des «porosités» d'éléments (non orientés) dans l'intervalle de direction K, avec K = 1 à 18; la porosité d'un élément est définie comme le produit ELLEN x W; (m^2).

DVEL(K) (NF,NT) : somme des vitesses d'écoulement dans les éléments à l'intérieur de l'intervalle de direction K, avec K = 1 à 36; (m/s).

DVM(K) (NT) : vitesse d'écoulement moyenne (non pondérée) dans tous les éléments à l'intérieur de l'intervalle de direction K, avec K = 1 à 36; (m/s).

DWTVM(K) (NT) : vitesse d'écoulement moyenne pondérée dans tous les éléments à l'intérieur de l'intervalle de direction K, avec K = 1 à 36; (m/s). Le facteur de pondération de la vitesse dans chaque élément est le produit ELLEN x VEL (voir DLVEL).

E(N) (NF) : conductance de l'élément N, définie comme la conductivité de l'élément x (section transversale/longueur), c'est-à-dire (W^2 x FLUID) x [(W x 1)/ELLEN]; (m^2/s).

EFF(L) (NW) : «efficacité» de l'intersection L; variable logique prenant la valeur (.TRUE.) lorsque l'intersection L peut contribuer à l'écoulement et la valeur (.FALSE.) dans le cas contraire.

EFFLE(I) (NW) : longueur totale des segments «efficaces» de la famille de fractures I; (m).

EFFVO(I) (NW) : «volume» total (c'est-à-dire ELLEN x AP x 1) des segments «efficaces» de la famille de fractures I; (m^3).

ELLEN(N) (NW,AP,NF) : longueur de l'élément N; (m).

ELOR(N) (NW,AP,NF) : angle de l'élément N à l'intérieur du plan de simulation; même convention que pour ATH(I).

ELORP(N) (NW,AP) : angle entre le plan de simulation et l'élément N; même convention que pour ATHP(I).

ESTNFR (NW) : nombre estimé de fractures requises pour la famille présentement traitée.

FLUID (NF) : constante égale au poids volumique [F/L^3] divisé par (12 x viscosité dynamique [FT/L^2]); (m^{-1} s^{-1}).

FR (NF) :	débit dans l'élément traité; positif dans le sens des y grandissants, en raison de la convention adoptée pour définir les éléments dans le sous-programme ELTDEF du programme NETWRK; (m^3/s).
I (NW,AP) :	indice des familles de fractures; équivalent à JSET.
I (NW,NF,NT) :	également indice pour les segments limites pour les nombres aléatoires uniformes R (NT).
IAAP (NW,AP) :	compteur du nombre de valeurs différentes du paramètre de distribution AAP.
IAP (NW,AP) :	compteur du nombre de réalisations de distributions d'ouvertures avec la même valeur de AAP.
IB(M) (NW,NF) :	code de position du noeud M; (0) pour un noeud interne, (1) à (8) pour un noeud situé sur le segment limite correspondant.
IBC1 (NT) :	condition aux limites pour le segment limite 1; égal à IBC(1).
IDIFF (NF) :	largeur de bande de la matrice A.
IDIR(J) (NT) :	direction (1 à 36) que doit prendre une particule pour la valeur correspondante de J aléatoire (J maximum est égal à 100).
INC1(L,1) (NW) :	numéro de famille de la première fracture du noeud L.
INC1(L,2) (NW) :	numéro d'individu de la première fracture du noeud L; remarquer que cette fracture appartient à la famille INC1(L,1).
INC1(L,1) (NW) :	numéro de famille de la deuxième fracture du noeud L.
INC2(L,2) (NW) :	numéro d'individu de la deuxième fracture du noeud L; remarquer que cette fracture appartient à la famille INC2(L,1); remarquer aussi que INC2(L,1 et 2) = 0 pour les noeuds appartenant à des limites.
ISET(N) (NW,AP,NF) :	famille de fractures qui contient l'élément N.

J (NW) : indice pour le numéro de fracture; équivalent à JFRA.

JO (NW) : nombre de fractures comprises entièrement dans la marge tampon (X1(J,I) = 999.) rencontrées jusqu'à maintenant dans la famille présentement traitée.

JAP (NF) : compteur du nombre de réalisations de distributions d'ouvertures; la valeur de JAP est à l'intérieur de l'intervalle 1 à (NAP x NAAP).

JFRA et JJFRA (NW) : indice pour le numéro de fracture; équivalent à J (dans le sous-programme CIRBDR).

JMEM(M) (NF) : nombre de noeuds reliés au noeud M ($1 \leq$ JMEM(M) ≤ 4).

JOINT(MAXNOD) (NF) : tableau de travail utilisé dans le sous-programme OPTNUM de renumérotage des noeuds (voir aussi NEWJT).

JSET (NW) : indice pour la famille de fractures; équivalent à I.

JT(2xMAXELT) (NF) : tableau simple équivalent aux deux tableaux NOD1 (MAXELT) et NOD2 (MAXELT) placés l'un à la suite de l'autre; JT est utilisé dans SETUP et dans OPTNUM seulement.

K (NF,NT) : indice de l'intervalle de direction; la valeur de K est comprise entre 1 et 18 ou entre 1 et 36 (degrés ÷ 10). K désigne l'intervalle de direction de 10(K - 1) à 10K.

KDIAG (MAXFRE) (NF) : tableau secondaire requis pour le stockage par largeur de bande variable de A; il donne l'adresse de tous les éléments diagonaux de la sous-matrice A.

L (NW) : indice des intersections de fractures; équivalent à LINT.

LC(MAXNOD) (NF) : code de condensation utilisé pour la décomposition des matrices des charges et de conductances aux noeuds; LC(I) précise le nombre de noeuds contraints qui ont un indice inférieur ou égal à I.

LINT (NW) :	indice des intersections de fractures; équivalent à L.
M (NW,NF) :	indice des noeuds, ou intersections «efficaces».
MAXA (NF) :	nombre maximum d'éléments de la matrice A.
MAXCON (NF) :	nombre maximum de noeuds contraints (c'est-à-dire avec une charge donnée) pour lesquels on a prévu de l'espace de mémoire (MAXCON = MAXNOD - MAXFRE).
MAXELT (NW,AP,NF) :	nombre maximum d'éléments pour lesquels on a prévu de l'espace de mémoire.
MAXFRA (NW) :	nombre maximum de fractures pour lesquelles on a prévu de l'espace de mémoire pour une famille de fractures.
MAXFRE (NF) :	nombre maximum de noeuds libres (c'est-à-dire sans charge définie) pour lesquels on a prévu de l'espace de mémoire.
MAXINT (NW) :	nombre maximum d'intersections pour lesquelles on a prévu de l'espace de mémoire.
MAXNOD (AP,NF) :	nombre maximum de noeuds (ou intersections «efficaces») pour lesquels on a prévu de l'espace de mémoire.
MEMJT(4(M-1)+1 à 4(M-1)+4) (NF) :	numéros des noeuds reliés au noeud M : leur nombre varie de 1 à 4 (voir JMEM); la dimension totale de MEMJT est 4 x MAXNOD.
MEND (NW) :	paramètre indiquant quelle extrémité (1 ou 2) est coupée dans la trace de fracture présentement traitée; paramètre utilisé seulement dans les sous-programmes CHOPXY, CHOPC et BNODE.
N (NW,AP,NF) :	indice des éléments.
NAPER (NF,NT) :	nombre total de réalisations différentes de distributions d'ouvertures; NAPER=NAP x NAAP.

NBN (NW) :	nombre de noeuds appartenant à une limite pour la famille de fractures présentement traitée.
NBO (NW,NF) :	nombre de segments limites différents du modèle, NBO vaut 6 pour un modèle circulaire et 8 pour un modèle rectangulaire.
ND1 et ND2 (AP) :	numéro du premier noeud et numéro du deuxième noeud de l'élément présentement traité (voir NOD1 et NOD2).
NEWJT(MAXNOD) (NF) :	tableau de travail utilisé dans le sous-programme OPTNUM de renumérotage des noeuds (voir aussi JOINT).
NEWNOD(M) (NF) :	après le renumérotage des noeuds, ce paramètre indique l'ancien numéro de noeud correspondant au nouveau noeud M (voir NODNUM).
NFRAC(I) (NW) :	nombre de fractures de la famille I à la fin du processus de génération.
NOD1(N) & NOD2(N) (NW,AP,NF) :	numéro du premier noeud (plus faible valeur de y) et numéro du deuxième noeud (plus grande valeur de y) de l'élément N (voir aussi JT).
NODNUM(M) (NF) :	après le renumérotage des noeuds, ce paramètre indique le nouveau numéro de noeud correspondant à l'ancien noeud M (voir NEWNOD).
NSEG(K) (NF,NT) :	nombre d'éléments de l'intervalle de direction K, avec K=1 à 36.
NT ((NT) :	indice pour la particule suivie.
NUMELT (NW,AP,NF) :	nombre d'éléments du modèle.
NUMINT (NW) :	nombre total d'intersections de fractures du modèle, y compris les intersections des fractures avec des limites.
NUMNPT (NW,AP,NF) :	nombre de noeuds, ou intersections «efficaces», du modèle.
P(N) (AP) :	premier cosinus directeur de l'élément N; voir aussi Q(N) et R(N); les conventions sont données à la figure D.2.

PHI(M) (NF) : valeur de la charge hydraulique au noeud M; (m). Dans le sous-programme SOLPHI, PHI est utilisé comme tableau de travail pour emmagasiner différents vecteurs.

PHIC(MAXCON) (NF) : vecteur des valeurs de charge hydraulique contrainte (c'est-à-dire définie).

Q(N) (AP) : deuxième cosinus directeur de l'élément N; voir aussi P(N) et R(N).

R(N) (AP) : troisième cosinus directeur de l'élément N; voir aussi P(N) et Q(N).

R(100) ((NT) : tableau de nombres aléatoires distribués uniformément.

RDFR(K) (NT) : rapport de la somme des débits dans l'intervalle de direction K, c'est-à-dire DFR(K), à la somme totale des débits dans toutes les directions.

RF (AP) : coordonnée radiale du point central de l'élément présentement traité, à partir du centre (0,0) du modèle; (m).

SPGR (NF) : poids volumique du fluide; (N/m^3).

SUMLE (NW) : somme des longueurs des traces de fractures générées jusqu'à maintenant pour la famille présentement traitée.

TANTH(J,I) (NW) : équivalent à TH(J,I).

TH(J,I) (NW) : angle de trace de la fracture J de la famille I, à l'intérieur du plan de simulation; même convention que pour ATH(I); exprimé sous forme de la tangente dans la plus grande partie du programme NETWRK et en radians dans les données de sortie.

THP(J,I) (NW) : angle entre le plan de simulation et la fracture J de la famille I; même convention que pour ATHP(I).

THR (AP) : coordonnée angulaire du point central de l'élément présentement traité; même convention que pour ATH(I).

TIM(NT) (NT) : temps de déplacement total nécessaire pour que la particule NT parcoure la distance donnée TRAVDIS(s).

TRACE(J) (NW) :	longueur de trace de la fracture J de la famille présentement traitée; (m).
TRAV(NT) (NT) :	distance absolue cumulative parcourue par la particule NT; (m).
VEL (NF) :	vitesse d'écoulement dans l'élément présentement traité; positive dans le sens des y grandissants, comme FR; (m/s).
VISC (NF) :	viscosité dynamique du fluide; (kg/(m.s)).
W(N) (NW,AP,NF) :	ouverture (largeur) de l'élément N; (m).
X1(J,I), Y1(J,I) (NW) :	coordonnées de l'extrémité 1 (plus faible valeur de y) de la fracture J de la famille I; (m). Remarquer que X1(J,I) = 999.0 lorsque la fracture (J,I) est comprise entièrement à l'intérieur de la marge tampon.
X2(J,I), Y2(J,I) (NW) :	coordonnées de l'extrémité 2 (plus grande valeur de y) de la fracture J de la famille I; (m).
X1E,Y1E et X2E,Y2E (AP) :	coordonnées des extrémités 1 (plus faible valeur de y) et 2 (plus grande valeur de y) de l'élément présentement traité; (m).
XC(J) et YC(J) (NW) :	coordonnées du point central de la trace de la fracture J de la famille présentement traitée; (m).
XORD(L, ou M) et YORD (L ou M) (NW,AP,NF) :	coordonnées de l'intersection L ou du noeud M; (m).
XTRAV(NT) et YTRAV(NT) (NT) :	distance totale parcourue par la particule NT dans la direction des x et dans la direction des y; (m).
Y1(J,I) (NW) :	voir X1(J,I).
Y2(J,I) (NW) :	voir X2(J,I).
Y1E et Y2E (AP) :	voir X1E.
YC(J) (NW) :	voir XC(J).
YORD(L ou M) (NW,AP,NF) :	voir XORD (L ou M).
YTRAV(NT) (NT) :	voir XTRAV(NT).

Annexe B
Fichiers d'entrée/sortie

Fichiers d'entrée/sortie

Tableau B.1. Liste de tous les fichiers d'entrée/sortie

Unité n°	Nom mnémonique	Description
1	INP1	Données d'entrée générales.
2	SUMNET	Données sommaires sur le réseau de lignes généré, les ouvertures de fractures (du programme NETWRK) et la porosité.
2+	SUMNET	Toutes les données susmentionnées et, en plus, des données sur les traces de fractures individuelles.
3	SPACNG1	Données d'espacement générées par le sous-programme SPCNG1.
4	SPCNG2	Données d'espacement générées par le sous-programme SPCNG2.
5	NODES	Fichier non mis en forme contenant des données relatives aux noeuds.
6	ELEM	Fichier non mis en forme contenant des données relatives aux éléments.
7	APER1	Fichier non mis en forme contenant des données relatives aux ouvertures générées par le programme NETWRK.
15	NODFO	Fichier mis en forme équivalent au fichier n° 5.
16	ELEFO	Fichier mis en forme équivalent au fichier n° 6.
17	APEF01	Fichier mis en forme équivalent au fichier n° 7.
21	INP2	Données d'entrée relatives à la distribution des ouvertures pour le programme APEGEN.

Tableau B.1 (suite)

Unité n°	Nom mnémonique	Description
22	APER2	Fichier non mis en forme contenant des données relatives aux ouvertures gérérées par le programme APEGEN.
23	APEF02	Fichier mis en forme équivalent au fichier n° 22.
31	SUMFLO	Résumé des calculs de paramètres d'écoulement, y compris les paramètres directionnels.
31+	SUMFLO	Données susmentionnées et diagrammes circulaires des débits.
31++	SUMFLO	Données susmentionnées et diagrammes circulaires de tous les paramètres directionnels.
32	DIRPAR	Fichier non mis en forme des paramètres directionnels.
33	FLOALL	Résultats des calculs des paramètres d'écoulement pour tous les éléments.
34	FLOMAT	Données détaillées sur le renumérotage des noeuds et les matrices des paramètres d'écoulement.
41	TRANSIT	Statistique et graphiques des temps de parcours.

Tableau B.2. Fichiers d'entrée et options de fichiers de sortie pour chaque programme

Programme	Unité d'entrée n°	Sortie	
		IPRT	Unité n°
NETWRK*	1	1	2,3,4,5,6,7
		2 ou 3	2+,3,4,5,6,7, 15,16,17
APEGEN	21,5,6,7	1	22
		2 ou 3	22,23
NETFLO	1,5,6,7**	1	31,32
		2	31+,32,33
		3	31++,32,33,34
NETRANS	1,32	1 ou 2 ou 3	41

* NETWRK génère aussi un fichier de données pour traçage de graphiques lorsque le paramètre d'entrée PLTSIZ est positif.

** Dans le programme NETFLO, l'unité d'entrée n° 7 doit correspondre à l'unité de sortie n° 7 du programme NETWRK (APER1) ou à l'unité de sortie n° 22 du programme APEGEN (APER2).

Annexe C
Structure des données d'entrée

Structure des données d'entrée

C.1. Fichier d'entrée général (unité 1)

- Enregistrement 1

  - Colonnes 1 à 80: TITRE (20A4).

- Enregistrement 2: données générales.

  - Colonnes 1 à 5: NSET (I5).
  - Colonnes 6 à 10: ISP (I5).
  - Colonnes 11 à 15: COEF (F5.0).
  - Colonnes 16 à 20: SEED (F5.0).
  - Colonnes 21 à 25: PLTSIZ (F5.0).
  - Colonnes 26 à 30: IGEOM (I5).
  - Colonnes 31 à 35: NAP (I5).
  - Colonnes 36 à 40: NAAP (I5).
  - Colonnes 41 à 50: DAAP (F10.0).
  - Colonnes 51 à 55: IPRT (I5).
  - Colonnes 56 à 60: NPART (I5).
  - Colonnes 61 à 65: TRAVDIS (F5.0).
  - Colonnes 66 à 68: NMAP (I3).

- Enregistrement 3: position des limites rectangulaires. Ces limites sont requises même pour un modèle circulaire.

  - Colonnes 1 à 40: XMINO, YMAXO, XMAXO, YMINO, XMINI, YMAXI, XMAXI, YMINI (8F5.0).
    (dans cet ordre dans le sens horaire)

- Enregistrement 4 et suivants ; autres données sur les limites du modèle. Un enregistrement pour chacune des NBO limites : pour un modèle rectangulaire, il y a 8 limites, pour un modèle circulaire, il y en a 6 (I = 1 à NBO).

  - Colonnes 1 à 5: ISHAP(I) (I5).
  - Colonnes 6 à 10: R(I) (F5.0).
  - Colonnes 11 à 15: XB(I) (F5.0).
  - Colonnes 16 à 20: YB(I) (F5.0).
  - Colonnes 21 à 25: IBC(I) (I5).
  - Colonnes 26 à 30: HDB(I) (F5.0).
  - Colonnes 31 à 35: HDE(I) (F5.0).

. Enregistrements suivants : première série de données sur le réseau de fractures. Un enregistrement pour chacune des NSET familles de fractures (I = 1 à NSET).

- Colonnes 1 à 10: DENS(I) (F10.0).
- Colonnes 11 à 15: LDSTR(I) (I5).
- Colonnes 16 à 20: THDSTR(I) (I5).
- Colonnes 21 à 25: APDSTR(I) (I5).
- Colonnes 26 à 30: ITERM(I) (I5).

. Enregistrements suivants : paramètres des distributions statistiques pour le réseau de fractures. Un enregistrement pour chacune des NSET familles de fractures (I = 1 à NSET).

- Colonnes 1 à 10: ALE(I) (F10.0).
- Colonnes 11 à 20: BLE(I) (F10.0).
- Colonnes 21 à 30: ATHP(I) (F10.0).
- Colonnes 31 à 40: BTHP(I) (F10.0).
- Colonnes 41 à 50: ATH(I) (F10.0).
- Colonnes 51 à 60: BTH(I) (F10.0).
- Colonnes 61 à 70: AAP(I) (F10.0).
- Colonnes 71 à 80: BAP(I) (F10.0).

. Dernier enregistrement :

Tableau des NMAP numéros de particules pour lesquels la trajectoire détaillée doit être imprimée (I = 1 à NMAP)

- Colonnes 1 à 30: MPART(I) (10I3).

C.2 Fichier d'entrée supplémentaire pour le programme APEGEN (unité 21)

. Enregistrement 1

- Colonnes 1 à 80: TITRE (20A4).
  (On recommande d'utiliser le même titre que celui du fichier d'entrée général et d'ajouter une identification pour l'exécution de génération d'ouvertures).

. Enregistrement 2: données générales.

- Colonnes 1 à 5: IPRT (I5).
- Colonnes 6 à 10: APDSTR (I5).
- Colonnes 11 à 15: NAP (I5).
- Colonnes 16 à 20: NAAP (I5).
- Colonnes 21 à 30: DAAP (F10.0).
- Colonnes 31 à 35: SEED (F5.0).
- Colonnes 36 à 40: NSET (I5).
- Colonnes 41 à 45: IOR (I5).

- Enregistrement 3 et suivants : paramètres des distributions d'ouvertures. Un enregistrement pour chacune des NSET familles de fractures (I = 1 à NSET).

  - Colonnes 1 à 10: AAP(I) (F10.0).
  - Colonnes 11 à 20: BAP(I) (F10.0).

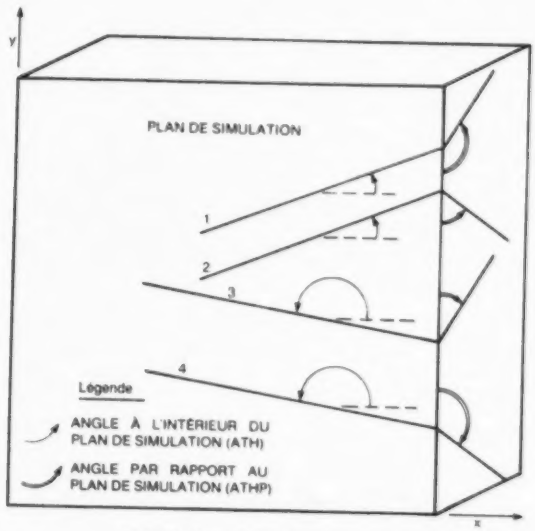
Annexe D

Conventions pour les angles, les directions, et la configuration des limites

Conventions pour les angles, les directions, et la configuration des limites

D.1 Conventions pour les angles

La figure D.1 illustre la convention adoptée pour exprimer les angles à l'intérieur du plan de simulation (ATH, TH et ELOR) et les angles par rapport au plan de simulation (ATHP, THP et ELORP).



DONNÉES D'ORIENTATION

FRACTURE	ATH (degrés)	ATHP (degrés)
1	20	135
2	20	45
3	170	45
4	170	135

Figure D.1. Conventions pour les angles.

D.2 Conventions pour les cosinus directeurs calculés dans le programme APEGEN

Le programme APEGEN permet de générer des valeurs d'ouvertures qui sont fonction de l'orientation des fractures (Section 2.9). A cette fin, le programme APEGEN calcule les cosinus directeurs de chaque segment de fracture (ou élément d'écoulement) en utilisant les conventions représentées à la figure D.2.

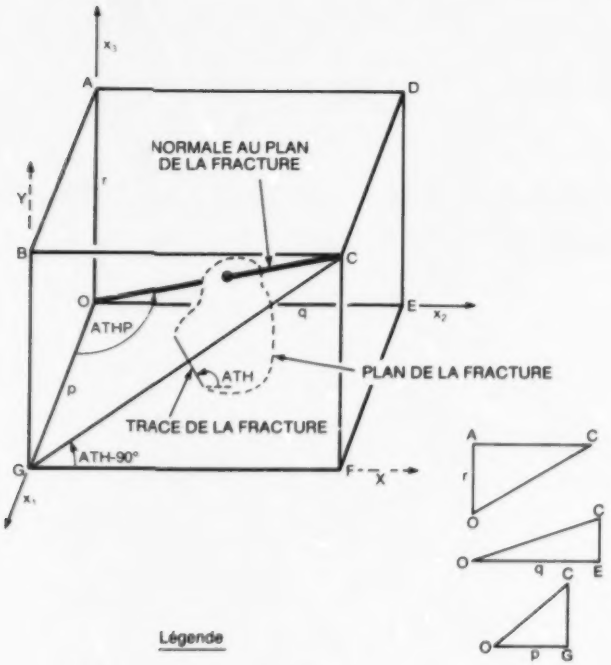


Figure D.2. Conventions pour les cosinus directeurs calculés dans le programme APEGEN.

Les angles TH et THP de la fracture de la figure D.2 correspondent aux angles ATH et ATHP respectivement de la figure D.1, et aux angles ELOR et ELORP pour les segments de fractures dans le programme APEGEN.

Dans la figure D.2, OC est le vecteur unitaire normal au plan de la fracture. Les trois cosinus directeurs sont définis comme suit :

$$p = \cos GOC = OG$$

$$q = \cos EOC = OE = BC$$

$$r = \cos AOC = OA = CF$$

Nous devons maintenant exprimer p, q et r en fonction des angles TH et THP. D'après la figure D.2, p est simplement cos THP. De plus, nous savons que :

$$\sin BGC = BC/GC$$

et

$$\sin GOC = GC$$

alors,
$$q = BC = GC \sin BGC = \sin GOC.\sin BGC$$
$$q = \sin THP.\sin(90-(TH-90))$$
$$q = \sin THP.\sin(180-TH)$$
$$q = \sin THP.\sin TH$$

Nous savons également que :

$$\sin CGF = CF/GC$$

alors, $r = CF = GC \sin CGF = \sin GOC . \sin CGF$

$r = \sin THP . \sin (TH-90)$

$r = -\sin THP . \cos TH$

En résumé :

$p = \cos THP$

$q = \sin THP . \sin TH$

$r = -\sin THP . \cos TH.$

D.3 Conventions et restrictions pour la configuration des limites

1. Chaque côté non circulaire d'un modèle peut être divisé en deux segments ayant chacun un type différent de condition aux limites.

2. Les limites d'écoulement sont numérotées dans le sens horaire, à partir de la limite verticale du côté gauche d'un modèle (figure D.3). Un modèle rectangulaire doit avoir 8 limites d'écoulement, un modèle circulaire doit en avoir 6. Dans un modèle rectangulaire, les limites d'écoulement portant les numéros 1, 3, 5 et 7 sont toujours requises (c'est-à-dire ISHAP(I) > 0). Les limites d'écoulement portant les numéros 2, 4, 6 et 8 peuvent avoir une longueur nulle (c'est-à-dire ISHAP(I)=0). Dans un modèle circulaire, les limites d'écoulement obligatoires sont celles portant les numéros 1, 3, 4 et 6; les limites optionnelles sont

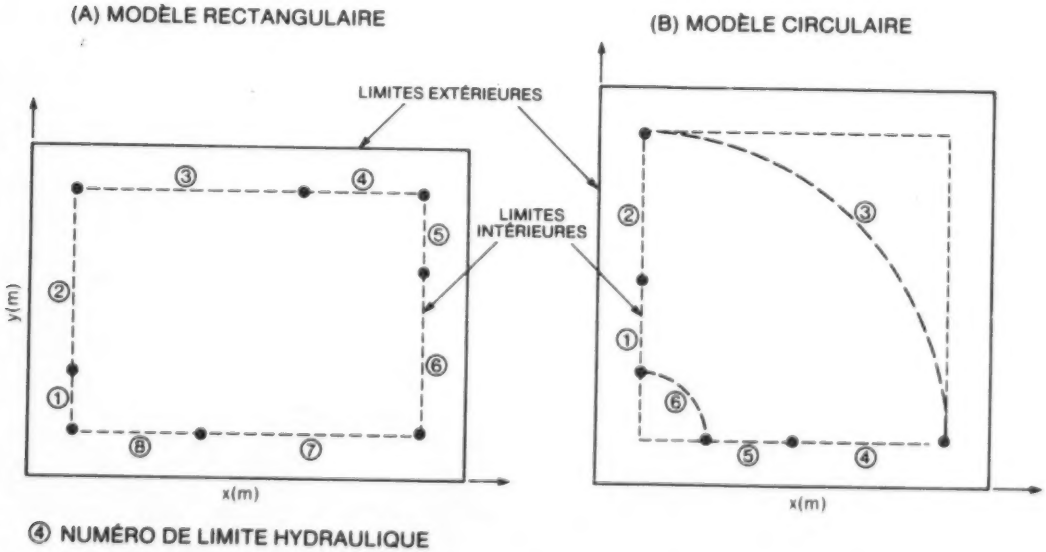


Figure D.3. Numérotage des limites d'écoulement : (A) modèle rectangulaire, (B) modèle circulaire.

celles portant les numéros 2 et 5. Remarquer que lorsque la limite d'écoulement I a une longueur nulle (c'est-à-dire ISHAP(I)=0), la limite d'écoulement numérotée I - 1 doit s'appliquer à la longueur entière du côté du modèle sur lequel elle se trouve.

3. L'orientation moyenne d'entrée (ATH) doit être à l'intérieur de l'intervalle 0° à 180°.

4. Toutes les limites rectangulaires (c'est-à-dire XMAXI...etc., et XMAXO...etc.) doivent être incluses dans les données d'entrée même dans le cas d'un modèle circulaire.

5. Une limite circulaire doit correspondre au quadrant supérieur droit d'un cercle et être centrée sur le point (XMINI, YMINI).

6. La version actuelle du programme NETRANS n'est applicable qu'aux modèles rectangulaires.

Annexe E
Dimensionnement des tableaux

Dimensionnement des tableaux

Dans les quatre prochaines pages, on reproduit le début des quatre programmes (NETWRK, APEGEN, NETFLO et NETRANS) et du sous-programme TTIMER du programme NETRANS. Les flèches indiquent les instructions auxquelles ont peut apporter des modifications si on désire effectuer un nouveau dimensionnement des tableaux. Remarquer que le programme NETRANS (et le sous-programme TTIMER) ne devrait nécessiter aucune modification des dimensions des tableaux (section 4.2.3). Cependant, si on veut changer le nombre maximum de particules, il faut changer la valeur de MAXNPRT dans le programme principal (NETRANS), de même que la dimension des tableaux suivants dans le sous-programme TTIMER: XTRAV, YTRAV, TIM, NSTEP et TRAV.

```
      PROGRAM NETWRK
C*****************************************************************************
C
C  PROGRAM TO GENERATE A TWO-DIMENSIONAL NETWORK OF FINITE LINE SEGMENTS
C  AND TO COMPUTE SEVERAL STATISTICS ON THE GENERATED LINE NETWORK.
C  (WRITTEN BY A. ROULEAU, 1983; UPDATE BY A. ROULEAU, JAN. 1986)
C
C*****************************************************************************
C
      DOUBLE PRECISION SEED,PI,RAD
      INTEGER THDSTR,APDSTR
C
C DIMENSION NSET = 4 , AND MAXFRA = 100
      DIMENSION THP(100,4)
      DIMENSION X1(100,4),Y1(100,4),X2(100,4),
     *Y2(100,4),TH(100,4),TANTH(100,4)
      INTEGER CUT1(100,4),CUT2(100,4)
C
C DIMENSION MAXFRA = 100
      DIMENSION XC(100),YC(100),TRACE(100),DEGRE(100),
     *DEGREP(100),APER(100)
C
C DIMENSION MAXINT=150; 2*MAXINT=300 (SEE DATA STATEMENT)
      DIMENSION XORD(150),YORD(150)
      INTEGER INC1(150,2),INC2(150,2),
     *IB(150)
      LOGICAL EFF(150)
C
C DIMENSION MAXELT = 200
      DIMENSION ELOR(200),ELORP(200),W(200),ELLEN(200)
      INTEGER NOD1(200),NOD2(200),ISET(200)
C
      COMMON /AREA1A/ XMINO,YMAXO,XMAXO,YMINO
      COMMON /AREA1B/ XMINI,YMAXI,XMAXI,YMINI,IGEOM
      COMMON /AREA2/ NSET,MAXFRA,MAXINT,MAXELT,NUMINT,NUMNPT,NUMELT,
     *NOINT,IPRT
      COMMON /AREA3A/ SEED,PI,RAD,COEF
      COMMON /AREA3B/ TITLE(20)
      COMMON /AREA4/ ATH(5),BTH(5),ATHP(5),BTHP(5),THDSTR(5),NFRAC(5)
      COMMON /AREA5/ DENS(5),ALE(5),BLE(5),LDSTR(5),ITERM(5)
      COMMON /AREA6/ AAP(5),BAP(5),APDSTR(5),DAAP,NAAP,IAAP,NAP,IAP
      COMMON /AREA7/ R(8),XB(8),YB(8),ISHAP(8),IBC(8)
C
      EQUIVALENCE (TH,TANTH),(DEGRE,APER),(TRACE,EFF,NOD1)
      EQUIVALENCE (IB,NOD2),(W,XC)
C
C DIMENSION MAXINT=150  (2*MAXINT=300)
      DATA IB/150*0/,INC2/300*0/
C
C ASSIGN VALUES TO SOME CONSTANTS
      MAXFRA=100
      MAXINT=150
      MAXELT=200
      PI=3.1415926536
      RAD=PI/180.
```

```
      PROGRAM APECEN
C*****************************************************************************
C
C  PROGRAM APECEN GENERATES FRACTURE APERTURE VALUES FOR THE FRACTURE
C  NETWORK GENERATED BY THE PROGRAM NETWRK
C
C  WRITTEN BY A. POULEAU, DECEMBER 1984; UPDATE BY A. POULEAU
C  AND R. MACLEOD, JUNE 1985.
C
C*****************************************************************************
      INTEGER APDSTR
      DOUBLE PRECISION SEED
      COMMON /AREA1/ TITLE(20)
      COMMON /AREA2/ NUMNPT,NUMELT,IPRT
      COMMON /AREA3/ AAP(5),BAP(5),SEED,APDSTR,NSET
      COMMON /AREA4/ DAAP,NAAP,NAP,IAAP,IAP
C
C DIMENSION MAXELT=200  <----
      INTEGER NOD1(200),NOD2(200),ISET(200)  <----
      DIMENSION W(200),WW(200),ELLEN(200),ELOR(200),ELORP(200)  <----
      DIMENSION P(200),Q(200),R(200)  <----
C
C DIMENSION MAXNOD=150  <----
      DIMENSION XORD(150),YORD(150)  <----
C
C CONSTANTS
      MAXELT=200  <----
      MAXNOD=150  <----
C
      READ(21,3) TITLE
3     FORMAT(20A4)
C
```

```
      PROGRAM NETFLO
C*****************************************************************
C
C  THE PROGRAM NETFLO IS A NUMERICAL CODE THAT SOLVES THE STEADY-
C  STATE EQUATIONS FOR FLUID FLOW IN THE NETWORK OF FRACTURES GENERATED
C  BY THE PROGRAM NETWRK FORTRAN.
C
C  WRITTEN BY A. ROULEAU, 1983; UPDATE BY A. ROULEAU AND D. LENTZ,
C  APRIL 1985.
C
C*****************************************************************
C
      COMMON /AREA2/ MAXNOD,MAXELT,NUMPNT,NUMELT,NC,NF,NA,MAXA,
     *MAXCON,MAXFRE,IPRT,JAP
      COMMON /AREA4/ XB(8),YB(8),HDB(8),HDE(8),ISHAP(8),IBC(8),IGEOM,NBO
      COMMON /AREA5/ MAXEL2,MAXNO4,IDIFF
      COMMON /AREA6/ TITLE(20)
      COMMON /AREA8/ VISC,SPGR
C
C DIMENSION = MAXELT = 200
      INTEGER NOD1(200),NOD2(200),IDUMMY(200)
      DIMENSION W(200),E(200),ELLEN(200),ELOR(200)
C
C DIMENSION = MAXEL2 = 2 * MAXELT = 400
      INTEGER JT(400)
C
C DIMENSION = MAXNOD = 150
      INTEGER IB(150),LC(150),JMEM(150),NODNUM(150),NEWNOD(150),
     *JOINT(150),NEWJT(150)
      DIMENSION XORD(150),YORD(150),PHI(150)
      LOGICAL REOR(150)
C
C DIMENSION = MAXNO4 = 4 * MAXNOD = 600
      INTEGER MEMJT(600)
C
C DIMENSION = MAXFRE = 110
      INTEGER KDIAG(110)
C
C DIMENSION = MAXCON = MAXNOD - MAXFRE = 150-110=40
      DIMENSION PHIC(40)
C
C DIMENSION = MAXA = 2000
      DIMENSION A(2000)
C
C MAXELT + 1 = 200 + 1 = 201
      EQUIVALENCE (JT(1),NOD1),(JT(201),NOD2)
      EQUIVALENCE (ELOR,IDUMMY)
C
C CONSTANTS
C
      MAXELT = 200
      MAXNOD = 150
      MAXA = 2000
      MAXFRE = 110
      MAXCON = MAXNOD-MAXFRE
```

```
      PROGRAM NETRANS
C*********************************************************************
C
C  PROGRAM TO SIMULATE STOCHASTICALLY THE MIGRATION OF PARTICLES
C  THROUGH A FRACTURE NETWORK BASED ON STATISTICS OF THE DIRECTIONAL
C  PARAMETERS
C
C WRITTEN BY A. ROULEAU AND D. LENTZ, 1984; UPDATE BY A.ROULEAU, JANUARY 1986.
C
C*********************************************************************
      COMMON /AREA1/ COSTH(36),SINTH(36),DLM(36),DWTVM(36),PDFR(36),
     +XLEN,YLEN,TRAVDIS,IDIR(100),MPART(10),IBC1,IPRT,JAP
      COMMON/AREA2/TITLE
      DIMENSION DVEL(36),DFR(36),DLE(36),DLVEL(36),DLVEL2(36),NSEG(36)
      CHARACTER TITLE(20)*4,FMT(5)*20
      DATA (FMT(I),I=1,5)/'(2(/),10I3)','(4(/),10I3)','(6(/),10I3)',
     +'(8(/),10I3)','(10(/),10I3)'/
      MAXNPRT=50     <--------
C


C*********************************************************************
C
C     SUBROUTINE TTIMER COMPUTES STOCHASTICALLY THE TRANSIT TIME OF
C  PARTICLES THROUGH THE NETWORK USING THE STATISTICS OF THE
C  DIRECTIONAL PARAMETERS (RECTANGULAR MODEL)
C
C*********************************************************************
      SUBROUTINE TTIMER(NPART,NM)
      COMMON /AREA1/ COSTH(36),SINTH(36),DLM(36),DWTVM(36),PDFR(36),
     +XLEN,YLEN,TRAVDIS,IDIR(100),MPART(10),IBC1,IPRT,JAP
      COMMON/AREA2/TITLE
      DIMENSION R(100),XTRAV(50),YTRAV(50),TIM(50)     <--------
     +,NSTEP(50),TRAV(50)     <--------
      DOUBLE PRECISION DSEED
      CHARACTER TITLE(20)*4,BCD1*22
C
```

Annexe F
Exemples d'exécution

Exemples d'exécution

Dans la présente annexe, on reproduit les fichiers d'entrée et des parties des fichiers de sortie de trois exemples d'exécution des quatre programmes **NETWRK**, **APEGEN**, **NETFLO** et **NETRANS**. Le premier exemple (réalisation 990) montre un modèle rectangulaire dans lequel aucune fracture ne se termine sur d'autres fractures. Dans la réalisation 991, on utilise une configuration circulaire. La réalisation 992 montre un autre modèle rectangulaire, dont les familles 2, 3 et 4 comprennent des fractures aboutissant sur d'autres fractures.

F.1 Modèle rectangulaire dont les fractures ne se terminent pas sur d'autres fractures : Réalisation 990

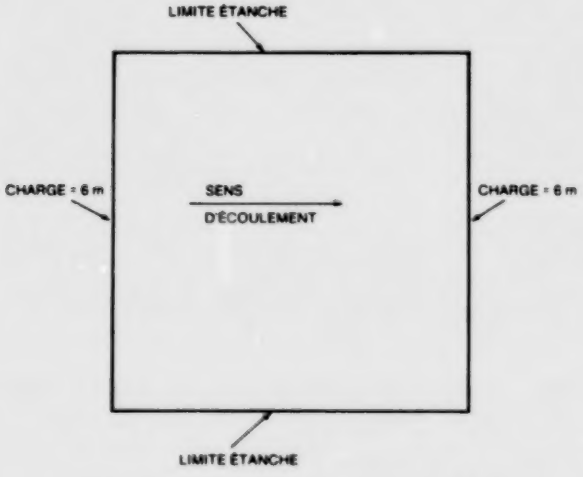


Figure F.1. Conditions aux limites de la réalisation 990.

Fichier INP1 - unité 1

```
990 SAMPLE RUN OF SDF PACK. : RECTANGULAR MODEL
   4     1  3.0 990.   25.     1     1     1                2   50 100.   0
-1.0 11.0 11.0 -1.0  0.0 10.0 10.0  0.0
   1        0.0  0.0    1  6.0
   0
   1        0.0 10.0    0
   0
   1       10.0 10.0    1  2.0
   0
   1       10.0  0.0    0
   0
     0.40    2    2    1    0
     0.60    2    2    1    0
     0.80    2    2    1    0
     1.00    2    2    1    0
      1.4       0.80      15.0       10.0      165.0       10.0  0.000005       0.0
      1.2       0.80      65.0       10.0      125.0       10.0  0.000005       0.0
      1.0       0.80     125.0       10.0       65.0       10.0  0.000005       0.0
      0.8       0.80     165.0       10.0       15.0       10.0  0.000005       0.0
 1 13 44
```

Fichier INP2 - unité 21

```
R990-1 FIRST SET OF APER. DATA FOR SAMPLE RUN ON RECT. MODEL
   2    2    1    3        0.5 990.    4    0
    -13.0       0.8
    -12.0       0.8
    -11.0       0.8
    -10.0       0.8
```

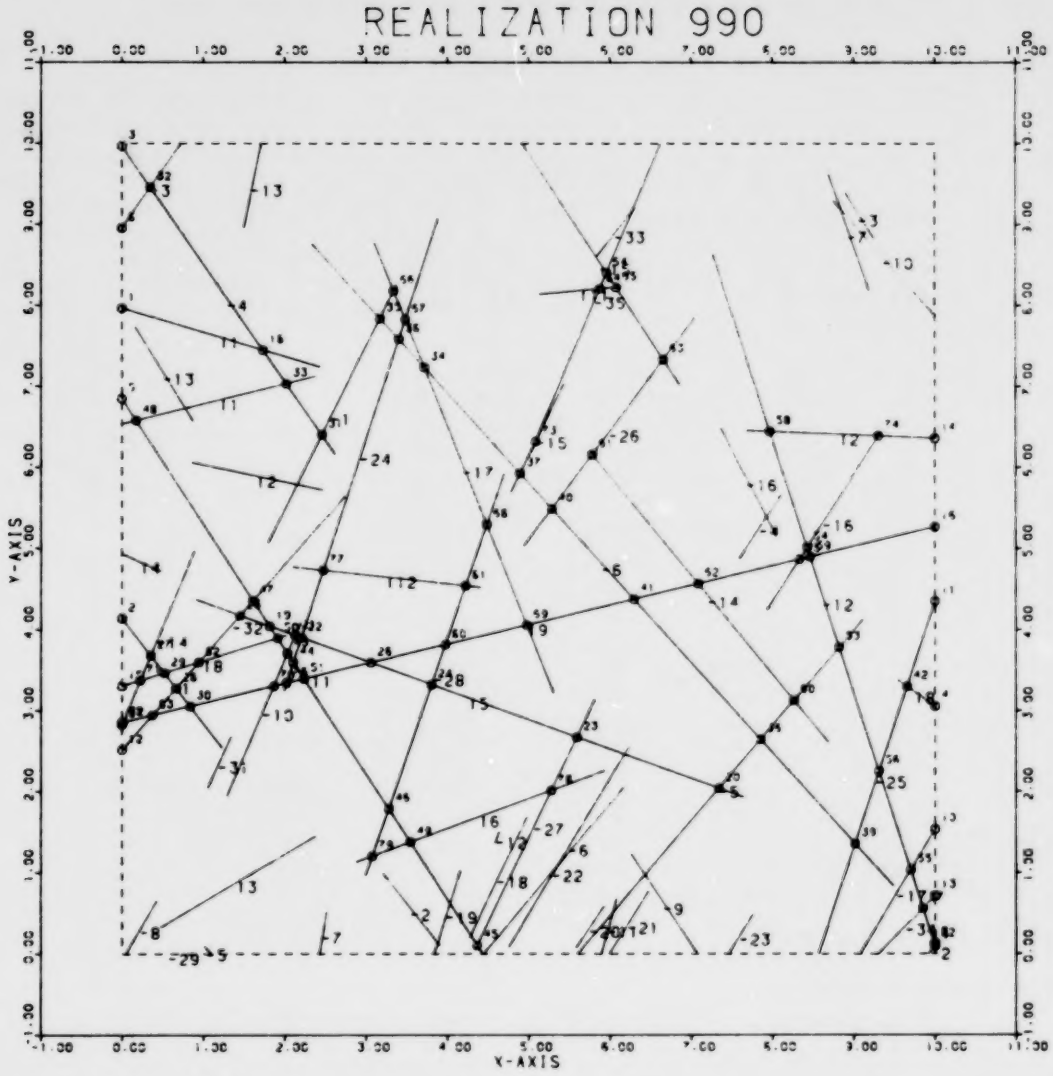


Figure F.2. Graphique tracé par ordinateur pour la réalisation 990; les numéros en petits caractères désignent les noeuds, les numéros en caractères plus gros désignent les fractures mais sans indiquer la famille; ces numéros ne sont pas imprimés lorsque NUMNPT >200 ou IPRT = 1.

Fichier SUMNET - unité 2

P990 SAMPLE RUN OF SDF PACK. : RECTANGULAR MODEL
FILE: SUMNET.U02

INPUT BOUNDARY DATA

NUMBER	SHAPE	RADIUS	XB	YB	IBC
1	1	.00	.000	.000	1
2	0	.00	.000	.000	0
3	1	.00	.000	10.000	0
4	0	.00	.000	.000	0
5	1	.00	10.000	10.000	1
6	0	.00	.000	.000	0
7	1	.00	10.000	.000	0
8	0	.00	.000	.000	0

INPUT DATA FOR FRACTURE NETWORK

FRACTURE SET :	1	2	3	4
DENSITY :	.400	.600	.800	1.000
DISTRIBUTION TYPE				
FOR TRACE LENGTH :	2	2	2	2
FOR ORIENTATION :	2	2	2	2
DISTRIBUTION PARAMETERS				
FOR TRACE LENGTH				
A :	1.40	1.20	1.00	.80
B :	.80	.80	.80	.80
FOR ORIENTATION WITHIN SIMULATION PLANE				
A :	165.00	125.00	65.00	15.00
B :	10.00	10.00	10.00	10.00
FOR ORIENTATION FROM SIMULATION PLANE				
A :	15.00	65.00	125.00	165.00
B :	10.00	10.00	10.00	10.00

FRACTURES OF SET 1

NUMBER GNRTD	NUMBER ACTUAL	LENGTH	ANGLE	X1	Y1	X2	Y2
	1	2.543	163.437	.000	7.965	2.437	7.240
	2	1.644	168.086	.864	6.060	2.473	5.720
	3	3.094	167.057	999.000	.691	-.788	.000
	4	.528	155.608	.000	4.934	.481	4.716
	5	7.160	160.073	.909	4.372	7.640	1.932

Fichier SUMNET - unité 2 (suite)

```
DENSITY OF FRACTURING       .1035 L/L**2
THEORETICAL MEAN TRACE LENGTH     5.5845
SURFACE AREA OF THE MODEL  100.0000
DESIRED TOTAL LENGTH FOR THE SET    10.3528
ACTUAL TOTAL LENGTH FOR THE SET     11.8748
ESTIMATED NUMBER OF FRACTURES   5
ACTUAL NUMBER OF FRACTURES GENERATED     5
NUMBER OF FRACTURES TOTALLY IN THE BUFFER MARGIN     1
NUMBER OF INTERSECTIONS WITH A BOUNDARY    2
```

FRACTURES OF SET 2

NUMBER GNRTD	ACTUAL	LENGTH	ANGLE	X1	Y1	X2	Y2
1		2.027	127.667	.000	4.140	1.239	2.535
2		1.135	128.303	3.206	.994	3.909	.103
3		.657	122.213	8.897	9.389	9.248	8.833
4		4.519	124.604	.000	9.967	2.623	6.165
5		.139	132.405	1.009	.103	1.103	.000
6		10.676	132.051	2.337	8.760	9.488	.833
7		1.525	109.211	8.691	9.627	9.192	8.187
8		.571	144.857	9.533	3.386	10.000	3.058
9		1.483	123.618	6.299	1.235	7.080	.000
10		1.888	131.471	8.750	9.293	10.000	7.878
11		8.161	122.922	.000	6.850	4.435	.000
12		8.970	107.750	7.265	8.640	10.000	.097
13		1.370	121.505	.166	7.735	.882	6.567
14		4.629	129.336	5.796	6.189	8.690	2.609
15		3.593	123.007	4.923	10.000	6.858	7.021
16		1.465	117.205	7.362	6.493	8.032	5.190
17		5.980	111.680	3.101	8.776	5.310	3.219

SUMMARY OF FRACTURE SET 2

```
DENSITY OF FRACTURING       .5438 L/L**2
THEORETICAL MEAN TRACE LENGTH     4.5722
SURFACE AREA OF THE MODEL  100.0000
DESIRED TOTAL LENGTH FOR THE SET    54.3785
ACTUAL TOTAL LENGTH FOR THE SET     58.8479
ESTIMATED NUMBER OF FRACTURES  35
ACTUAL NUMBER OF FRACTURES GENERATED   17
NUMBER OF FRACTURES TOTALLY IN THE BUFFER MARGIN     0
NUMBER OF INTERSECTIONS WITH A BOUNDARY  10
```

FRACTURES OF SET 3

NUMBER GNRTD	ACTUAL	LENGTH	ANGLE	X1	Y1	X2	Y2

. . . etc

Fichier SUMNET - unité 2 (fin)

APERTURE AND POROSITY DATA

INPUT DATA FOR APERTURE DISTRIBUTIONS IAAP= 1

FRACTURE SET :	1	2	3	4
DISTRIBUTION TYPE :	1	1	1	1
DISTRIBUTION PARAMETERS				
A :	5.00E-06	5.00E-06	5.00E-06	5.00E-06
B :	.00	.00	.00	.00

NUMBER OF ELEMENTS (NUMELT) : 116

SUMMARY OF POROSITY DATA IAAP= 1 IAP= 1

SET	LENGTH		VOLUME	
	EFFECTIVE	DEAD	EFFECTIVE	DEAD
1	8.67	3.81	4.03E-05	1.90E-05
2	39.18	19.67	1.96E-04	9.83E-05
3	31.58	37.66	1.58E-04	1.88E-04
4	20.53	5.47	1.03E-04	2.73E-05

********** //// END OF LIST ////
********** //// END OF LIST ////

Fichier SPACING 1 - unité 3

R990 SAMPLE RUN OF SDF PACK. : RECTANGULAR MODEL
FILE: SPACING1.U03

FRACTURE SET 1

NUMB	NTEN	LINE	XP	YP	FRACT	SPACING
1	1	7	.25	1.1[illegible]	2	1.08
2	1	7	.25	1.19	5	1.00
3	2	4	1.77	4.76	5	1.14
4	2	8	1.81	4.13	5	1.15
5	4	6	1.13	9.66	4	1.89
6	5	7	1.39	4.62	2	1.10
7	5	7	1.39	4.62	5	.99
8	5	8	3.11	9.81	2	1.12
9	5	8	3.11	9.81	5	.98
10	5	9	1.97	9.85	2	1.02
11	6	4	3.03	9.66	2	1.12
12	6	4	3.03	9.66	5	.98
13	6	6	1.85	9.67	2	1.03
14	6	8	.49	5.26	4	1.92
15	7	4	1.78	8.67	2	1.02
16	7	5	1.39	8.53	4	1.93
17	8	1	.50	5.82	4	1.91
18	9	5	2.97	7.99	5	1.17
19	9	6	2.66	9.93	2	1.02
20	9	8	.88	7.76	4	1.90
21	9	10	.90	4.73	2	1.01
22	10	2	.64	2.69	2	1.08
23	10	2	.64	2.69	5	1.00
24	10	10	1.40	9.16	4	1.91
25	12	8	.34	.96	2	1.09
26	12	8	.34	.96	5	.99
27	13	1	1.39	4.94	2	1.09
28	13	1	1.39	4.94	5	.99
29	13	2	.99	1.31	5	1.15
30	14	6	.30	1.88	2	1.07
31	14	6	.30	1.88	5	1.00
32	14	8	.44	5.52	4	1.92
33	14	10	.95	2.27	2	1.11
34	14	10	.95	2.27	5	.98
35	15	5	.11	2.97	2	1.02
36	15	7	.90	4.15	2	1.07
37	15	7	.90	4.15	5	1.01
38	15	8	3.12	9.32	5	1.15
39	15	9	1.47	3.65	5	1.15
40	15	10	2.73	7.68	5	1.15
41	16	10	1.12	5.25	2	1.00
42	17	4	.08	.44	2	1.08
43	17	4	.08	.44	5	1.00
44	18	1	.46	.34	2	1.12
45	18	1	.46	.34	5	.98
46	18	3	.68	5.66	4	1.89
47	18	8	2.89	9.01	2	1.12
48	18	8	2.89	9.01	5	.98
49	18	9	1.44	2.36	5	1.16
50	21	1	1.97	7.67	2	1.08

. . . etc

Fichier NODFO - unité 15

```
R990 SAMPLE RUN OF SDF PACK. : RECTANGULAR MODEL
FILE: NODFO.U15


                        INTERSECTION DATA

NUMBER OF INTERSECTIONS
          EFFECTIVE :            84 (NUMNPT)
          NON-EFFECTIVE :        34 (NOINT)
          MAXINT=               150

FILE CONTAINS :
          XORD(NUMNPT), FORMAT(1X,F10.6)
          YORD(NUMNPT), FORMAT(10F10.6)
          IB(NUMNPT), FORMAT(40I2)
          INC1(MAXINT,2), FORMAT(20I5)
          INC2(MAXINT,2), FORMAT(20I5)
  .000000    .000000    .000000 10.000000    .000000 10.000000 10.000000    .000000    .000000 10.000000
10.000000    .000000 10.000000 10.000000    .000000 10.000000    .000000   1.737363   1.818023   7.337093
 2.131146   2.206437   5.591844   3.803644   1.454607   3.057432    .354742    .666739    .521605    .836477
 2.459313    .351168   2.022156   3.713952   3.168090   7.852433   4.893169   3.399670   9.013976   5.285862
 6.293326   9.654153   2.036488   2.107275   4.365382   3.276245   1.614642    .176458   3.542914   1.914441
 2.237254   9.991733   8.820281   8.425710   9.699223   9.309294   9.850472   7.968618   8.462836   8.263844
 5.782113   7.086290   6.656292   5.958018   6.074981   3.337118   3.478614   4.483662   4.977173   1.867637
  .226592    .011057   5.084364   9.301488   8.331626   2.025632   2.475746   5.277655   3.075242   3.972881
 4.222935    .946643    .376193   5.874889
 7.965372   4.139865   9.966771   3.057648   6.850134    .096961    .136448   8.953638   2.830361   1.537404
 4.350513   2.516386    .716847   6.361823   3.303467   5.270401   2.854183   7.448657   4.042244   2.041434
 3.928728   3.901434   2.674133   3.322403   4.173992   3.592925   3.680333   3.276172   3.464179   3.056294
 6.402271   9.457792   7.035881   7.234345   7.839501   2.646322   5.927036   7.582766   1.358607   5.491686
 4.374787   3.301105   3.704830   3.595501    .107912   1.790057   4.356360   6.577598   1.378192   3.893328
 3.394752    .122788   3.782347   5.014971   1.036575   2.254695    .564080   6.442903   4.898988   3.129221
 6.158089   4.566385   7.332004   8.406950   8.226892   8.182250   7.826324   5.298179   4.056776   3.305444
 3.373283   2.856854   6.320194   6.389703   4.867285   3.343619   4.732300   2.016731   1.206047   3.814117
 4.540642   3.595138   2.945079   8.208392
1 1 1 5 1 5 5 1 1 5 5 1 5 5 1 5 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   1    2    2    2    2    2    3    3    3    3    3    3    3    4    4    4    4    1    1    1
   1    1    1    1    1    1    2    2    2    2    2    2    2    2    2    2    2    2    2    2
   2    2    2    2    2    2    2    2    2    2    2    2    2    2    2    2    2    2    2    2
   2    2    2    2    2    2    2    2    2    3    3    3    3    3    3    3    3    3    3    3
   3    3    3    3    3    2    2    3    2    3    3    3    2    3    3    2    1    2    3    2
   3    3    3    2    2    1    3    2    2    3    3    3    3    3    1    3    3    2    0    0
                                                                                          . . . etc
```

Fichier ELEFO - unité 16

```
R990 SAMPLE RUN OF SDF PACK. : RECTANGULAR MODEL
FILE: ELEFO.U16


                        ELEMENT DATA

NUMBER OF ELEMENTS :            116 (NUMELT)

FILE CONTAINS :
          NOD1(NUMELT), FORMAT(20I5)
          NOD2(NUMELT), FORMAT(20I5)
          ISET(NUMELT), FORMAT(50I2)
          ELLEN(NUMELT), FORMAT(1P10E10.3)
          ELOR(NUMELT), FORMAT(1P10E10.3)
          FLORP(NUMELT), FORMAT(1P10E10.3)
   18   20   23   24   26   22   21   19   30   28   29   27   31   33   18   32   39   36   41   40
   37   34   38    4   45   49   46   51   44   43   50   19   47   48    6   52   57   55   56   53
   59   54   60   62   63   65   69   68   34   67   31   35   52    8   20   36   60   70   43    9
   72   71   37   75   54   55   76   44   22   77   38   39   56   42   40   61   45   78   79   46
   24   80   81   12   83   28   82   25   57   73   84   48   14   74   79   49   15   71   29   82
   17   72   83   30   70   76   51   26   80   69   41   62   75   59   84   81
    1   23   24   26   22   21   19   25   28   29   27    2   33   18   32    3   36   41   40   37
   34   38   35   42   49   46   51   44   43   50   19   47   48    5   52   57   55   56   53   59
   54   58   62   61   65   64   68   34   67   66   35   66    7   32   36   60   53   43   21   72
   71   27   73   54   74   10   44   22   77   38   67   56   42   11   61   63   78   23   46   24
   80   81   68   83   28   82   25   47   13   84   64   33   74   58   49   78   71   29   82   50
   72   83   30   70   76   51   26   80   69   41   62   75   59   16   65   77
1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
1.813E+00 1.856E+00 1.902E+00 7.937E-01 9.052E-01 8.009E-02 3.331E-01 3.866E-01 2.778E-01 2.375E-01
2.731E-01 5.805E-01 7.698E-01 5.015E-01 2.441E+00 6.184E-01 1.734E+00 2.328E+00 1.504E+00 5.863E-01
1.761E+00 4.692E-01 3.457E-01 4.229E-01 1.513E+00 4.907E-01 1.912E+00 2.392E-01 1.302E-01 2.246E-01
1.774E-01 3.742E-01 2.646E+00 3.247E-01 2.712E-02 4.634E-01 4.961E-01 1.279E+00 1.604E+00 1.172E+00
1.218E-01 1.499E+00 1.858E+00 2.058E+00 1.067E+00 2.147E-01 1.336E+00 2.084E+00 6.370E-01 3.830E-01
1.602E+00 3.822E-01 1.597E-02 6.144E-01 7.946E-01 6.344E-01 8.580E-01 4.336E-01 2.431E-01 2.871E-02
5.596E-01 3.327E-01 4.372E-01 1.751E-01 1.630E+00 5.842E-01 2.648E-01 3.216E-01 8.734E-01 2.996E+00
2.560E-01 9.435E-01 1.102E+00 1.105E+00 8.309E-01 1.464E+00 2.116E+00 7.286E-01 6.176E-01 1.621E+00
5.200E-01 7.684E-01 8.011E-01 5.703E-01 4.405E-01 4.244E-01 7.701E-01 2.426E-01 2.138E-01 2.047E+00
2.153E-01 1.902E+00 6.991E-01 1.334E+00 4.983E-01 1.849E+00 2.371E-01 3.087E-01 4.448E-01 1.013E+00
1.138E-02 3.756E-01 4.735E-01 1.061E+00 1.625E-01 2.177E-01 8.438E-01 9.418E-01 1.033E+00 1.354E+00
8.158E-01 1.281E+00 1.350E-01 1.581E+00 2.009E-01 1.758E+00
  163.437   160.073   160.073   160.073   160.073   160.073   160.073   160.073   127.667   127.667
  127.667   127.667   124.604   124.604   124.604   124.604   132.051   132.051   132.051   132.051
  132.051   132.051   132.051   144.857   122.922   122.922   122.922   122.922   122.922   122.922
```

. . . etc

Fichier APEF01 - unité 17

```
R990 SAMPLE RUN OF SDF PACK. : RECTANGULAR MODEL
FILE: APEF01.U17


                        FRACTURE APERTURES

NUMBER OF ELEMENTS :              116 (NUMELT)
NUMBER OF DIFF. VAL. FOR FIRST PARAM (AAP) :        1 (NAAP)
NUMBER OF REALIZATIONS FOR EACH VAL. OF AAP :       1 (NAP)

FILE CONTAINS : NAAP*NAP ARRAYS W(NUMELT), FORMAT(1P10E10.3)
5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06
5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06
5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06
5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06
5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06
5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06
5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06
5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06
5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06
5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06
5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06
5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06
5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06 5.000E-06

************                //// END OF LIST ////
************                //// END OF LIST ////
```

Fichier APEFO2 - unité 23

R990-1 FIRST SET OF APER. DATA FOR SAMPLE RUN ON RECT. MODEL
FILE : APEFO2.U23

FRACTURE APERTURES

NUMBER OF ELEMENTS : 116 (NUMELT)
DISTRIBUTION TYPE : 2

INPUT DATA FOR APERTURE DISTRIBUTIONS IAAP= 1 IAP= 1

FRACTURE SET :	1	2	3	4
DISTRIBUTION PARAMETERS				
A :	-1.30E+01	-1.20E+01	-1.10E+01	-1.00E+01
B :	.80	.80	.80	.80

ARRAY ISET, FORMAT(50I2)

```
1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
```

ARRAY WW, FORMAT(1P10E10.3)

```
1.726E-06 7.523E-06 2.515E-06 1.228E-06 2.350E-06 3.985E-07 3.890E-06 9.165E-07 2.006E-06 9.368E-06
8.109E-06 7.618E-06 1.782E-06 9.002E-06 2.603E-06 1.817E-05 7.023E-06 2.228E-05 3.259E-06 1.517E-05
1.325E-05 4.516E-06 8.818E-06 9.216E-06 1.063E-05 2.053E-06 6.752E-06 1.304E-05 1.868E-06 1.790E-06
1.915E-06 1.194E-05 1.469E-05 4.938E-06 1.086E-05 5.760E-06 4.494E-06 1.027E-05 2.059E-05 4.277E-06
5.322E-06 5.745E-06 5.090E-06 1.272E-05 3.250E-05 3.372E-06 5.441E-05 3.988E-06 2.632E-05 5.011E-06
2.590E-05 1.493E-05 4.256E-06 2.056E-05 3.502E-05 8.458E-06 2.299E-05 6.069E-06 3.106E-05 5.392E-05
4.313E-05 7.731E-06 1.462E-05 5.176E-06 2.201E-05 1.064E-05 1.250E-05 4.324E-06 1.096E-05 1.167E-05
1.384E-05 1.519E-05 2.143E-05 1.485E-05 6.136E-06 2.164E-05 3.612E-05 2.459E-05 3.176E-05 1.930E-05
1.414E-05 4.918E-06 4.021E-06 1.529E-05 2.656E-05 9.868E-06 8.824E-06 3.273E-05 2.198E-05 1.103E-05
1.458E-05 3.011E-05 3.233E-05 3.125E-05 9.171E-05 7.124E-06 3.541E-05 1.554E-05 4.352E-05 2.865E-05
8.896E-05 6.721E-05 9.277E-05 5.282E-05 4.236E-05 7.810E-05 1.609E-04 2.749E-04 3.715E-05 3.626E-05
2.572E-05 4.230E-05 1.320E-04 3.044E-05 8.677E-05 3.222E-05
```

. . . etc

Fichier SUMFLO - unité 31

```
R990 SAMPLE RUN OF SCF PACK. : RECTANGULAR MODEL
FILE: SUMFLO.U31

          OUTPUT FROM SUBROUTINE BOUND


          TOTAL NUMBER OF NODES              84 (NUMNPT)
          NUMBER OF FREE NODES               67 (NF)
          NUMBER OF CONSTRAINED NODES        17 (NC)


 BDRY NO    IBC        BEG          END         'GRADIENT'

      1      1          .000       10.000          .000
      2      0          .000         .000          .000
      3      0          .000       10.000          .000
      4      0          .000         .000          .000
      5      1          .000         .000          .000
      6      0          .000         .000          .000
      7      0          .000         .000          .000
      8      0          .000         .000          .000

          OUTPUT FROM SETUP

INITIAL IDIFF=    71



          OUTPUT FROM OPTNUM

FINAL IDIFF=    10



          RESULTS FROM OPSTOR

SIZE OF MATRIX A (NA):     469
```

Fichier SUMFLO - unité 31 (suite)

SELECTED RESULTS OF FLOW CALCULATIONS FOR SEGMENTS LOCATED AT A BOUNDARY

BDRY.NO	ELEM.NO	WIDTH(M)	LENGTH(M)	VELOCITY(M/SEC)	FLOW RATE(M**3/SEC)
1	1	5.0000000E-06	1.8125744E+00	-2.3578586E-06	1.1789293E-11
1	12	5.0000000E-06	5.8052746E-01	-2.6959696E-06	1.3479848E-11
1	16	5.0000000E-06	6.1836823E-01	-7.7480072E-07	3.8740036E-12
5	24	5.0000000E-06	4.2294354E-01	-4.6465305E-06	-2.3232653E-11
1	34	5.0000000E-06	3.2467384E-01	-5.2352579E-06	2.6176290E-11
5	35	5.0000000E-06	2.7117867E-02	-3.0076542E-07	-1.5038271E-12
5	53	5.0000000E-06	1.5966943E-02	5.1081266E-07	-2.5540633E-12
1	54	5.0000000E-06	6.1440259E-01	7.7980165E-07	3.8990083E-12
1	60	5.0000000E-06	2.8708182E-02	1.9295495E-06	9.6477475E-12
5	66	5.0000000E-06	5.8420596E-01	2.8731184E-06	-1.4365592E-11
5	74	5.0000000E-06	1.1049289E+00	1.7089283E-06	-8.5446415E-12
1	84	5.0000000E-06	5.7034958E-01	3.3441906E-06	1.6720953E-11
5	89	5.0000000E-06	2.1376720E-01	1.7972861E-06	-8.9864306E-12
5	93	5.0000000E-06	6.9906866E-01	-7.0903772E-06	-3.5451886E-11
1	97	5.0000000E-06	2.3710406E-01	4.6448844E-06	2.3224422E-11
1	101	5.0000000E-06	1.1375387E-02	4.8696241E-06	2.4348120E-11
5	114	5.0000000E-06	1.5813977E+00	7.8861186E-06	-3.9430593E-11

Fichier SUMFLO - unité 31 (suite)

SUMMARY OF FLOW CALCULATIONS FOR JAP= 1

TOTAL FLOW ALONG THE BOUNDARIES

BDRY NO	FLOW (M**3/SEC) (- =>FLOW OUT)
1	1.33159E-10
2	.00000E+00
3	.00000E+00
4	.00000E+00
5	-1.33159E-10
6	.00000E+00
7	.00000E+00
8	.00000E+00

DIRECTIONAL POROSITY AND PERMEABILITY

DIRECTION (DEGR/10)	LENGTH (M)	POROSITY (M**2)	PERMEABILITY (M**3)
1	2.01E-01	1.00E-06	5.02E-12
2	1.42E+01	7.10E-05	3.55E-10
3	2.35E+00	1.17E-05	5.87E-11
4	.00E+00	.00E+00	.00E+00
5	4.95E+00	2.47E-05	1.24E-10
6	2.31E+00	2.66E-05	1.33E-10
7	9.13E+00	4.56E-05	2.28E-10
8	1.22E+01	6.10E-05	3.05E-10
9	.00E+00	.00E+00	.00E+00
10	.00E+00	.00E+00	.00E+00
11	6.66E+00	3.33E-05	1.67E-10
12	4.44E+00	2.22E-05	1.11E-10
13	1.89E+01	9.46E-05	4.73E-10
14	8.73E+00	4.36E-05	2.18E-10
15	4.23E-01	2.11E-06	1.06E-11
16	.00E+00	.00E+00	.00E+00
17	8.07E+00	4.03E-05	2.02E-10
18	3.79E+00	1.90E-05	9.48E-11

Fichier SUMFLO - unité 31 (suite)

OTHER DIRECTIONAL PARAMETERS

DIRECTION (DEGR/10)	NSEG	(1)VELOC (M/S)	(2)FL.RA (M**3/S)	(3)LENGTH (M)	(3)X(1)	(3)X(1)**2
1	1	8.35E-07	4.18E-12	2.01E-01	1.68E-07	1.40E-13
2	19	1.21E-04	6.03E-10	1.42E+01	9.68E-05	7.68E-10
3	2	3.91E-06	1.96E-11	2.35E+00	5.56E-06	1.40E-11
4	0	.00E+00	.00E+00	.00E+00	.00E+00	.00E+00
5	9	4.11E-05	2.06E-10	4.95E+00	2.71E-05	1.79E-10
6	6	1.76E-05	8.80E-11	3.85E+00	1.32E-05	5.28E-11
7	12	2.52E-05	1.26E-10	9.13E+00	2.24E-05	8.94E-11
8	11	2.93E-05	1.47E-10	1.08E+01	2.90E-05	1.28E-10
9	0	.00E+00	.00E+00	.00E+00	.00E+00	.00E+00
10	0	.00E+00	.00E+00	.00E+00	.00E+00	.00E+00
11	3	7.41E-06	3.71E-11	2.79E+00	8.00E-06	2.36E-11
12	0	.00E+00	.00E+00	.00E+00	.00E+00	.00E+00
13	0	.00E+00	.00E+00	.00E+00	.00E+00	.00E+00
14	0	.00E+00	.00E+00	.00E+00	.00E+00	.00E+00
15	0	.00E+00	.00E+00	.00E+00	.00E+00	.00E+00
16	0	.00E+00	.00E+00	.00E+00	.00E+00	.00E+00
17	0	.00E+00	.00E+00	.00E+00	.00E+00	.00E+00
18	0	.00E+00	.00E+00	.00E+00	.00E+00	.00E+00
19	0	.00E+00	.00E+00	.00E+00	.00E+00	.00E+00
20	0	.00E+00	.00E+00	.00E+00	.00E+00	.00E+00
21	0	.00E+00	.00E+00	.00E+00	.00E+00	.00E+00
22	0	.00E+00	.00E+00	.00E+00	.00E+00	.00E+00
23	0	.00E+00	.00E+00	.00E+00	.00E+00	.00E+00
24	1	1.23E-06	6.13E-12	1.46E+00	1.79E-06	2.20E-12
25	0	.00E+00	.00E+00	.00E+00	.00E+00	.00E+00
26	2	1.43E-16	7.13E-12	1.39E+00	9.10E-07	9.69E-13
27	0	.00E+00	.00E+00	.00E+00	.00E+00	.00E+00
28	0	.00E+00	.00E+00	.00E+00	.00E+00	.00E+00
29	5	1.40E-05	7.01E-11	3.87E+00	1.64E-05	7.93E-11
30	4	9.58E-06	4.79E-11	4.44E+00	8.98E-06	2.69E-11
31	22	7.14E-05	3.57E-10	1.89E+01	5.65E-05	2.10E-10
32	7	3.58E-05	1.79E-10	8.73E+00	4.52E-05	2.60E-10
33	1	4.46E-06	2.23E-11	4.23E-01	1.89E-06	8.43E-12
34	0	.00E+00	.00E+00	.00E+00	.00E+00	.00E+00
35	8	4.62E-05	2.31E-10	8.07E+00	4.29E-05	2.67E-10
36	3	1.30E-05	6.49E-11	3.79E+00	1.42E-05	6.31E-11

Fichier SUMFLO - unité 31 (fin)

ROSE OF FLOW RATES (M**3/S)

//// END OF LIST ////
//// END OF LIST ////

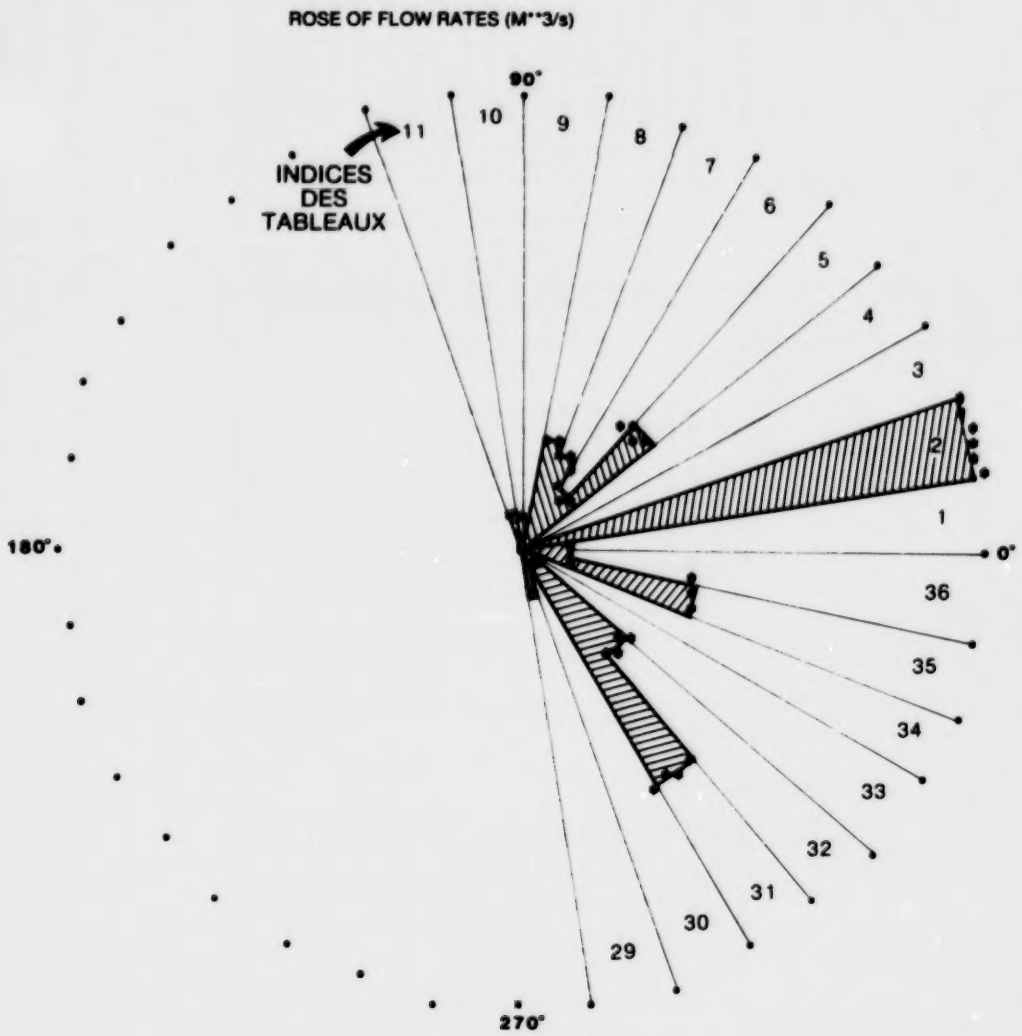


Figure F.3. Diagramme circulaire du fichier SUMFLO complété à la main.

Fichier FLOALL - unité 33

```
R990 SAMPLE RUN OF SDF PACK. : RECTANGULAR MODEL
FILE: FLOALL.U33

        RESULTS OF FLOW CALCULATIONS,  JAP=   1

        HEAD AT NODAL POINTS

        FORMAT(3(I   IB(I)   PHI(I)))

  1   1  6.0000000E+00     2   1  6.0000000E+00     3   1  6.0000000E+00     4   5  2.0000000E+00
  5   1  6.0000000E+00     6   5  2.0000000E+00     7   5  2.0000000E+00     8   1  6.0000000E+00
  9   1  6.0000000E+00    10   5  2.0000000E+00    11   5  2.0000000E+00    12   1  6.0000000E+00
 13   5  2.0000000E+00    14   5  2.0000000E+00    15   1  6.0000000E+00    16   5  2.0000000E+00
 17   1  6.0000000E+00    18   0  5.7902527E+00    19   0  5.4793321E+00    20   0  3.6608947E+00
 21   0  5.3714738E+00    22   0  5.3368393E+00    23   0  4.4447461E+00    24   0  4.8707362E+00
 25   0  5.5560360E+00    26   0  5.0879165E+00    27   0  5.9231895E+00    28   0  5.8299749E+00
 29   0  5.8683799E+00    30   0  5.7962460E+00    31   0  5.4560384E+00    32   0  5.9764863E+00
 33   0  5.6939598E+00    34   0  4.7899931E+00    35   0  4.9607461E+00    36   0  3.3253596E+00
 37   0  4.1488182E+00    38   0  4.8908655E+00    39   0  2.7432078E+00    40   0  3.9735566E+00
 41   0  3.7675485E+00    42   0  2.0926704E+00    43   0  5.3978811E+00    44   0  5.3634930E+00
 45   0  4.7307904E+00    46   0  4.8996332E+00    47   0  5.5464532E+00    48   0  5.9165804E+00
 49   0  4.8320554E+00    50   0  5.4697596E+00    51   0  5.3127774E+00    52   0  2.0004003E+00
 53   0  2.8055567E+00    54   0  2.6033181E+00    55   0  2.0823763E+00    56   0  2.4264833E+00
 57   0  2.0188556E+00    58   0  2.4127811E+00    59   0  2.6120505E+00    60   0  3.1499411E+00
 61   0  3.8389775E+00    62   0  3.3819204E+00    63   0  3.9270029E+00    64   0  3.9952934E+00
 65   0  3.9911807E+00    66   0  4.9198695E+00    67   0  4.8789011E+00    68   0  4.7210369E+00
 69   0  4.4793778E+00    70   0  5.4206777E+00    71   0  5.9459499E+00    72   0  5.9972814E+00
 73   0  4.1225256E+00    74   0  2.2432608E+00    75   0  2.6516954E+00    76   0  5.3716782E+00
 77   0  5.1269097E+00    78   0  4.5692202E+00    79   0  4.8622327E+00    80   0  4.8356400E+00
 81   0  4.8429077E+00    82   0  5.7392669E+00    83   0  5.9063916E+00    84   0  3.9994167E+00

        FLOW IN ELEMENTS,     JAP=   1

NUMBER    WIDTH(M)       LENGTH(M)     VELOC (M/SEC) FL.RATE(M**3/SEC) REYNOLD NO.     DIRECTION     TRANSIT TIME
  1  5.0000000E-06  1.8125744E+00 -2.3578586E-06 -1.1789293E-11  1.1765762E-08  1.6343680E+02  7.6873750E+05
  2  5.0000000E-06  1.8563956E+00 -8.6036011E-06 -4.3018006E-11  4.2932141E-08  1.6007307E+02  2.1576961E+05
  3  5.0000000E-06  1.9020807E+00 -4.5633917E-06 -2.2816958E-11  2.2771416E-08  1.6007307E+02  4.1681294E+05
  4  5.0000000E-06  7.9373426E-01 -5.5752250E-06 -2.7876125E-11  2.7820484E-08  1.6007307E+02  1.4236811E+05
  5  5.0000000E-06  9.0519125E-01 -5.6032684E-06 -2.8016342E-11  2.7960421E-08  1.6007307E+02  1.6154701E+05
  6  5.0000000E-06  8.0085488E-02 -8.8119639E-06 -4.4059819E-11  4.3971876E-08  1.6007307E+02  9.0882679E+03
                                                                                                    . . . etc
```

R990 SAMPLE RUN OF SDF PACK. : RECTANGULAR MODEL
FILE : TRANSIT.U41

AUXILIARY DIRECTIONAL PARAMETERS FOR JAP= 1

DIRECTION	RDFR	DLM	DWTVM
1	1.88E-03	2.01E-01	8.35E-07
2	2.72E-01	7.47E-01	7.93E-06
3	8.83E-03	1.17E+00	2.52E-06
4	.00E+00	.00E+00	.00E+00
5	9.28E-02	5.50E-01	6.48E-06
6	3.97E-02	6.42E-01	3.99E-06
7	5.69E-02	7.61E-01	3.98E-06
8	6.62E-02	9.87E-01	4.42E-06
9	.00E+00	.00E+00	.00E+00
10	.00E+00	.00E+00	.00E+00
11	1.67E-02	9.31E-01	2.95E-06
12	.00E+00	.00E+00	.00E+00
13	.00E+00	.00E+00	.00E+00
14	.00E+00	.00E+00	.00E+00
15	.00E+00	.00E+00	.00E+00
16	.00E+00	.00E+00	.00E+00
17	.00E+00	.00E+00	.00E+00
18	.00E+00	.00E+00	.00E+00
19	.00E+00	.00E+00	.00E+00
20	.00E+00	.00E+00	.00E+00
21	.00E+00	.00E+00	.00E+00
22	.00E+00	.00E+00	.00E+00
23	.00E+00	.00E+00	.00E+00
24	2.77E-03	1.46E+00	1.23E-06
25	.00E+00	.00E+00	.00E+00
26	3.22E-03	6.93E-01	1.66E-06
27	.00E+00	.00E+00	.00E+00
28	.00E+00	.00E+00	.00E+00
29	3.16E-02	7.74E-01	4.83E-06
30	2.16E-02	1.11E+00	3.00E-06
31	1.61E-01	8.60E-01	3.72E-06
32	8.08E-02	1.25E+00	5.76E-06
33	1.01F-02	4.23E-01	4.46E-06
34	.00E+00	.00E+00	.00E+00
35	1.04E-01	1.01E+00	6.23E-06
36	2.93E-02	1.26E+00	4.45E-06

ARRAY IDIR
FORMAT: 10(J IDIR(J))

1	2	2	2	3	2	4	2	5	2	6	2	7	2	8	2	9	2	10	2
11	2	12	2	13	2	14	2	15	2	16	2	17	2	18	2	19	2	20	2
21	2	22	2	23	2	24	2	25	2	26	2	27	2	28	3	29	5	30	5
31	5	32	5	33	5	34	5	35	5	36	5	37	5	38	6	39	6	40	6
41	6	42	7	43	7	44	7	45	7	46	7	47	7	48	8	49	8	50	8
51	8	52	8	53	8	54	8	55	11	56	11	57	29	58	29	59	29	60	30
61	30	62	31	63	31	64	31	65	31	66	31	67	31	68	31	69	31	70	31
71	31	72	31	73	31	74	31	75	31	76	31	77	31	78	32	79	32	80	32
81	32	82	32	83	32	84	32	85	32	86	33	87	35	88	35	89	35	90	35
91	35	92	35	93	35	94	35	95	35	96	35	97	36	98	36	99	36	100	0

SIZE OF GENERATED NETWORK :
X-LENGTH= 10.0000 (M)
Y-LENGTH= 10.0000 (M)

PARTICLES MUST TRAVEL 100.00 (M) IN THE X DIRECTION

Fichier TRANSIT - unité 41 (suite)

STATISTICS OF ALL PARTICLES FOR JAP= 1

NT	XTRAV(M)	YTRAV(M)	TRAV(M)	TIME(S)	NSTEP
1	1.00E+02	4.38E+00	1.44E+02	2.95E+07	169
2	1.00E+02	-3.38E+00	1.49E+02	3.12E+07	174
3	1.00E+02	4.87E+00	1.48E+02	2.99E+07	176
4	1.00E+02	4.91E+00	1.51E+02	3.17E+07	177
5	1.00E+02	-5.88E+00	1.45E+02	2.91E+07	173
6	1.00E+02	-1.39E+01	1.38E+02	2.68E+07	164
7	1.00E+02	-8.30E+00	1.51E+02	3.14E+07	178
8	1.00E+02	9.81E+00	1.40E+02	2.74E+07	171
9	1.00E+02	3.61E+00	1.53E+02	3.13E+07	181
10	1.00E+02	-1.69E+01	1.37E+02	2.74E+07	156
11	1.00E+02	-1.17E+01	1.52E+02	3.16E+07	175
12	1.00E+02	-1.95E+00	1.41E+02	2.77E+07	166
13	1.00E+02	8.32E-01	1.37E+02	2.59E+07	164
14	1.00E+02	4.97E+00	1.37E+02	2.57E+07	164
15	1.00E+02	2.38E+00	1.44E+02	2.90E+07	171
16	1.00E+02	1.46E+01	1.47E+02	2.93E+07	175
17	1.00E+02	-4.74E+00	1.47E+02	2.95E+07	168
18	1.00E+02	-9.15E+00	1.45E+02	2.89E+07	167
19	1.00E+02	-3.24E+00	1.42E+02	2.84E+07	167
20	1.00E+02	-2.31E+00	1.45E+02	2.90E+07	176
21	1.00E+02	6.34E+00	1.50E+02	3.03E+07	184
22	1.00E+02	-7.57E-01	1.43E+02	2.78E+07	171
23	1.00E+02	-6.46E+00	1.46E+02	2.97E+07	172
24	1.00E+02	-1.35E+00	1.46E+02	2.90E+07	174
25	1.00E+02	-2.66E+00	1.41E+02	2.77E+07	163
26	1.00E+02	-3.48E+00	1.45E+02	2.98E+07	170
27	1.00E+02	-4.82E+00	1.41E+02	2.81E+07	166
28	1.00E+02	-2.97E+00	1.38E+02	2.68E+07	164
29	1.00E+02	-1.61E+01	1.44E+02	2.90E+07	171
30	1.00E+02	5.19E+00	1.48E+02	2.94E+07	177
31	1.00E+02	4.20E-01	1.42E+02	2.78E+07	168
32	1.00E+02	2.95E+00	1.42E+02	2.78E+07	166
33	1.00E+02	4.25E+00	1.43E+02	2.77E+07	168
34	1.00E+02	-1.43E+01	1.41E+02	2.78E+07	165
35	1.00E+02	1.48E+00	1.42E+02	2.80E+07	170
36	1.00E+02	2.49E+00	1.33E+02	2.50E+07	158
37	1.00E+02	-1.34E+01	1.35E+02	2.61E+07	158
38	1.00E+02	-3.90E-01	1.41E+02	2.85E+07	172
39	1.00E+02	-1.21E+01	1.43E+02	2.88E+07	166
40	1.00E+02	-4.37E+00	1.42E+02	2.87E+07	163
41	1.00E+02	-4.58E+00	1.43E+02	2.91E+07	171
42	1.00E+02	-9.28E-01	1.46E+02	3.01E+07	173
43	1.00E+02	-1.20E+01	1.45E+02	3.09E+07	167
44	1.00E+02	-1.18E+01	1.39E+02	2.82E+07	162
45	1.00E+02	-5.55E+00	1.41E+02	2.84E+07	165
46	1.00E+02	-1.54E+01	1.47E+02	3.05E+07	168
47	1.00E+02	1.00E+01	1.43E+02	2.81E+07	173
48	1.00E+02	-8.98E+00	1.43E+02	2.82E+07	170
49	1.00E+02	-3.46E+00	1.49E+02	3.01E+07	174
50	1.00E+02	7.85E+00	1.40E+02	2.79E+07	175

BASIC DESCRIPTIVE STATISTICS OF ABS.TRAV.DIST.,TRAV(M)

NUMBER OF DATA POINTS	MEAN	STANDARD DEVIATION	MAXIMUM	MINIMUM
50	1.434E+02	4.390E+00	1.528E+02	1.328E+02

Fichier TRANSIT - unité 41 (fin)

FREQUENCY DIAGRAMS OF TRANSIT TIME (S)

BASIC DESCRIPTIVE STATISTICS OF TRANSIT TIME (S)

NUMBER OF DATA POINTS	MEAN	STANDARD DEVIATION	MAXIMUM	MINIMUM
50	2.872E+07	1.519E+06	3.173E+07	2.501E+07

MIDDLE INTERV	FREQ	CUM REL FREQ	RELATIVE FREQUENCY DIAGRAM	CUMULATIVE FREQUENCY DIAGRAM
2.52E+07	1	2	1*	1*
2.55E+07	0	2	1	1*
2.59E+07	2	6	1**	1***
2.62E+07	1	8	1*	1****
2.65E+07	0	8	1	1****
2.69E+07	2	12	1**	1******
2.72E+07	0	12	1	1******
2.75E+07	5	22	1*****	1**********
2.79E+07	6	34	1******	1******************
2.82E+07	4	42	1****	1**********************
2.85E+07	4	50	1****	1**************************
2.89E+07	5	60	1*****	1********************************
2.92E+07	5	70	1*****	1**************************************
2.95E+07	3	76	1***	1**
2.99E+07	2	80	1**	1**
3.02E+07	3	86	1***	1**
3.06E+07	1	88	1*	1**
3.09E+07	1	90	1*	1**
3.12E+07	3	96	1***	1**
3.16E+07	1	98	1*	1**
3.19E+07	1	100	1*	1**

********** //// END OF LIST ////

********** //// END OF LIST ////

F.2 Modèle rectangulaire avec fractures se terminant sur d'autres fractures : Réalisation 992

Toutes les données d'entrée sont les mêmes que pour la réalisation 990, à l'exception de la valeur de ITERM qui est 1 pour les familles de fractures 2, 3 et 4.

Fichier INP1 - unité 1

```
992 SAMPLE RUN OF SDF PACK. : RECTANG. MODEL WITH ABUTMENT
   4     1   3.0 990.   25.     1     1     1                2   50 100.   0
-1.0 11.0 11.0 -1.0  0.0 10.0 10.0  0.0
   1        0.0  0.0    1  6.0
   0
   1        0.0 10.0    0
   0
   1       10.0 10.0    1  2.0
   0
   1       10.0  0.0    0
   0
     0.40     2    2    1    0
     0.60     2    2    1    1
     0.80     2    2    1    1
     1.00     2    2    1    1
      1.4        0.90      15.0      10.0     165.0      10.0  0.000005      0.0
      1.2        0.80      65.0      10.0     125.0      10.0  0.000005      0.0
      1.0        0.90     125.0      10.0      65.0      10.0  0.000005      0.0
      0.8        0.80     165.0      10.0      15.0      10.0  0.000005      0.0
 1 13 44
```

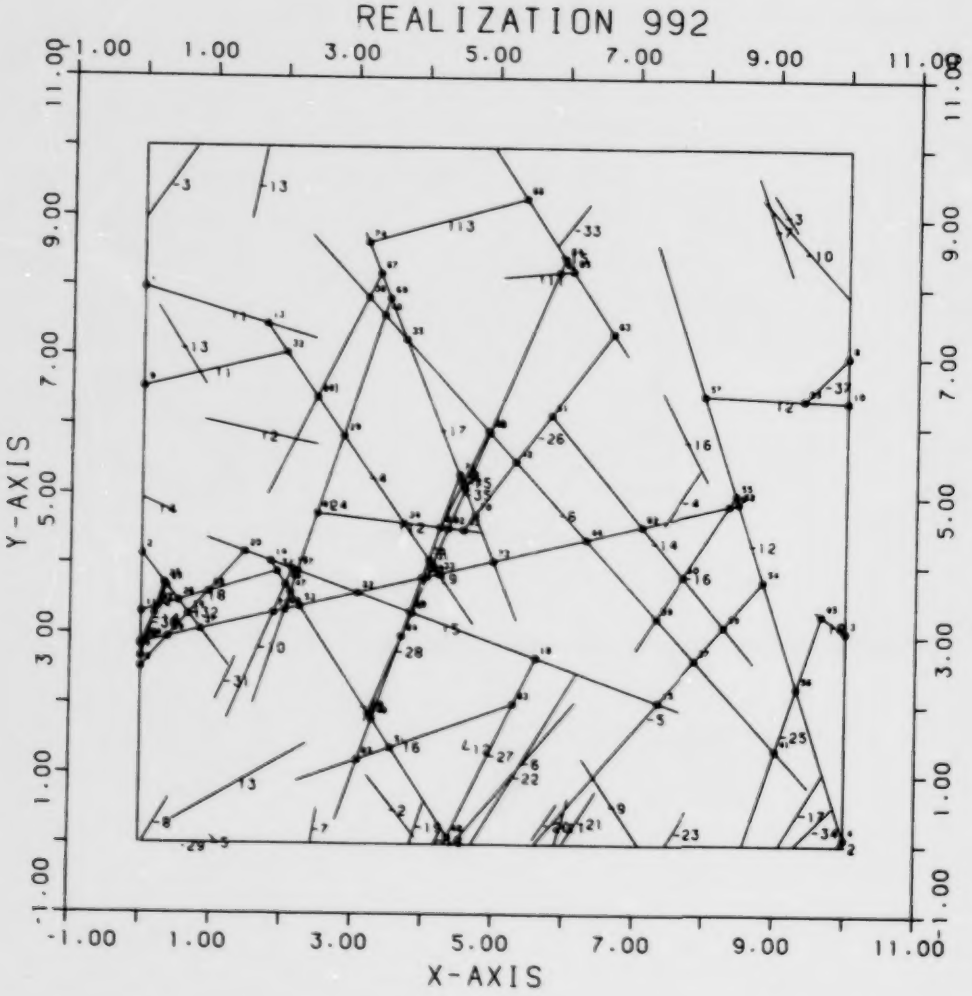


Figure F.4. Graphique tracé par ordinateur pour la réalisation 992; voir figure F.2 pour la signification des numéros.

F.3 Modèle circulaire : réalisation 991

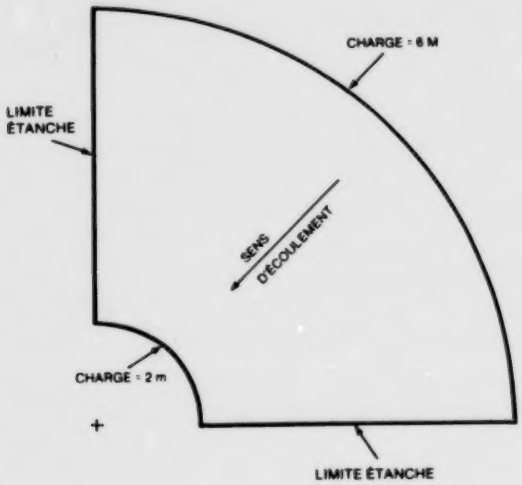


Figure F.5. Conditions aux limites pour la réalisation 991.

Fichier INP1 - unité 1

```
-991 SAMPLE RUN OF SDF PACK. : CIRCULAR MODEL
   4     2   6.0 991.   10.     2     1     1                    2
-1.0 11.0  11.0 -1.0   0.0 10.0 10.0   0.0
   1        0.0  2.5     0
   0
   2 10.0   0.0 10.0     1  6.0
   1       10.0  0.0     0
   0
   2  2.5   2.5  0.0     1  2.0
     0.40     2    2     1    0
     0.60     2    2     1    0
     0.80     2    2     1    0
     1.00     2    2     1    0
      1.4        0.80       15.0        10.0       165.0        10.0  0.000005
      1.2        0.90       65.0        10.0       125.0        10.0  0.000005
      1.0        0.80      125.0        10.0        65.0        10.0  0.000005
      0.8        0.90      165.0        10.0        15.0        10.0  0.000005
```

Fichier INP2 - unité 21

```
R991-1 FIRST SET OF APER. DATA FOR SAMPLE RUN ON CIRC. MODEL
   2    2    1    3        0.5 991.    4    0
    -13.0       0.8
    -12.0       0.8
    -11.0       0.8
    -10.0       0.8
```

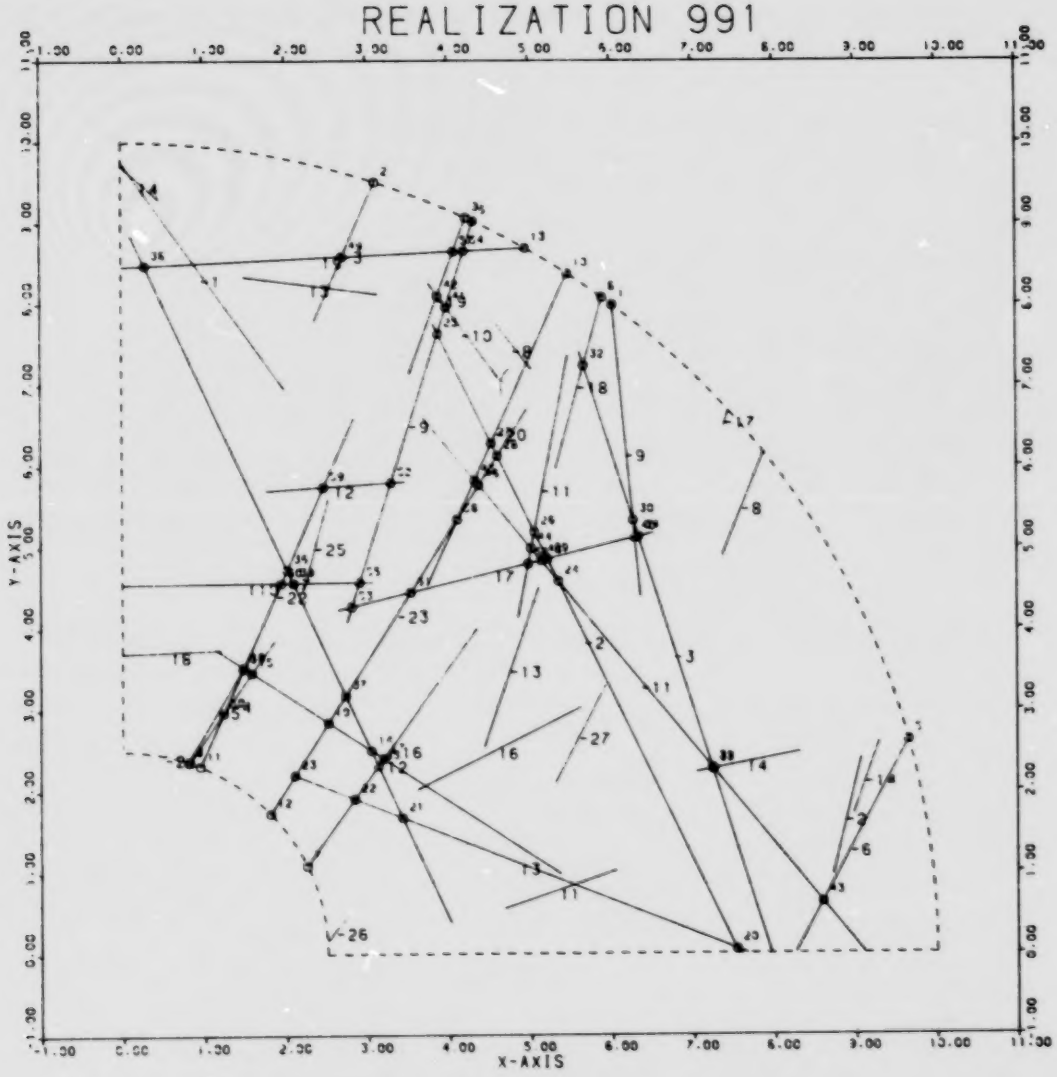


Figure F.6. Graphique tracé par ordinateur pour la réalisation 991; voir figure F.2 pour la signification des numéros.

Annexe G
Listes des programmes

Listes des programmes

G.1 Liste du programme NETWRK

```
      PROGRAM NETWRK
C*****************************************************************
C
C  PROGRAM TO GENERATE A TWO-DIMENSIONAL NETWORK OF FINITE LINE SEGMENTS
C  AND TO COMPUTE SEVERAL STATISTICS ON THE GENERATED LINE NETWORK.
C  (WRITTEN BY A. ROULEAU, 1983; UPDATE BY A. ROULEAU, JAN. 1986)
C
C*****************************************************************
C
      DOUBLE PRECISION SEED,PI,RAD
      INTEGER THDSTR,APDSTR
C
C DIMENSION NSET = 4 , AND MAXFRA = 100
      DIMENSION THP(100,4)
      DIMENSION X1(100,4),Y1(100,4),X2(100,4),
     *Y2(100,4),TH(100,4),TANTH(100,4)
      INTEGER CUT1(100,4),CUT2(100,4)
C
C DIMENSION MAXFRA = 100
      DIMENSION XC(100),YC(100),TRACE(100),DEGRE(100),
     *DEGREP(100),APER(100)
C
C DIMENSION MAXINT=150; 2*MAXINT=300 (SEE DATA STATEMENT)
      DIMENSION XORD(150),YORD(150)
      INTEGER INC1(150,2),INC2(150,2),
     *IB(150)
      LOGICAL EFF(150)
C
C DIMENSION MAXELT = 200
      DIMENSION ELCR(200),ELORP(200),W(200),ELLEN(200)
      INTEGER NOD1(200),NOD2(200),ISET(200)
C
      COMMON /AREA1A/ XMINO,YMAXO,XMAXO,YMINO
      COMMON /AREA1B/ XMINI,YMAXI,XMAXI,YMINI,IGEOM
      COMMON /AREA2/ NSET,MAXFRA,MAXINT,MAXELT,NUMINT,NUMNPT,NUMELT,
     *NOINT,IPRT
      COMMON /AREA3A/ SEED,PI,RAD,COEF
      COMMON /AREA3B/ TITLE(20)
      COMMON /AREA4/ ATH(5),BTH(5),ATHP(5),BTHP(5),THDSTR(5),NFRAC(5)
      COMMON /AREA5/ DENS(5),ALE(5),BLE(5),LDSTR(5),ITERM(5)
      COMMON /AREA6/ AAP(5),BAP(5),APDSTR(5),DAAP,NAAP,IAAP,NAP,IAP
      COMMON /AREA7/ R(8),XB(8),YB(8),ISHAP(8),IBC(8)
C
      EQUIVALENCE (TH,TANTH),(DEGRE,APER),(TRACE,EFF,NOD1)
      EQUIVALENCE (IB,NOD2),(W,XC)
C
C DIMENSION MAXINT=150  (2*MAXINT=300)
      DATA IB/150*0/,INC2/300*0/
C
C ASSIGN VALUES TO SOME CONSTANTS
      MAXFRA=100
      MAXINT=150
      MAXELT=200
      PI=3.1415926536
      RAD=PI/180.
```

```
C
C   READ IN DATA
C
      READ(1,3) TITLE
    3 FORMAT(20A4)
      READ(1,5)NSET,ISP,COEF,SEED,PLTSIZ,IGEOM,NAP,NAAP,DAAP,IPRT
    5 FORMAT(2I5,3F5.0,3I5,F10.0,I5)
      READ(1,7) XMINO,YMAXO,XMAXO,YMINO,XMINI,YMAXI,XMAXI,YMINI
    7 FORMAT(8F5.0)
C
C   CHECK FOR MIXED UP RECTANGULAR BOUNDARIES
      IF(XMINI .GT. XMINO .AND. XMAXI .LT. XMAXO .AND. YMINI .GT. YMINO
     *.AND. YMAXI .LT. YMAXO) GO TO 19
      WRITE(2,8)
    8 FORMAT(///,'1** ERROR IN INPUT DATA: MIXED RECTANGULAR BOUNDARIES')
      STOP
C
   19   IF(IGEOM .EQ. 1) NBO=8
        IF(IGEOM .EQ. 2) NBO=6
      READ(1,22)(ISHAP(I),R(I),XB(I),YB(I),IBC(I),I=1,NBO)
   22 FORMAT(I5,3F5.0,I5)
      READ(1,25)(DENS(I),LDSTR(I),THDSTR(I),APDSTR(I),ITERM(I),I=1,NSET)
   25 FORMAT(F10.0,4I5)
      READ(1,29) (ALE(I),BLE(I),ATHP(I),BTHP(I),
     *ATH(I),BTH(I),AAP(I),BAP(I),I=1,NSET)
   29 FORMAT(8F10.0)
      WRITE(2,34) TITLE
34    FORMAT('1',//,1X,20A4,//,' FILE: SUMNET.U02')
C
C PRINT OUT BOUNDARY DATA
C
      WRITE(2,70)
   70 FORMAT(///,1CX,'INPUT BOUNDARY DATA',///,' NUMBER SHAPE RADIUS ',
     *6X,'XB',9X,'YB',2X,'IBC',/)
      WRITE(2,71)(I,ISHAP(I),R(I),XB(I),YB(I),IBC(I),I=1,NBO)
   71 FORMAT(I5,I7,F8.2,2F10.3,I4)
C
C PRINT INPUT DATA FOR FRACTURE NETWORK
C
      WRITE(2,80)(I,I=1,NSET)
   80 FORMAT(///,1CX,'INPUT DATA FOR FRACTURE NETWORK',///,' FRACTURE ',
     *'SET :',T25,5I10)
      WRITE(2,85)(DENS(I),I=1,NSET)
   85 FORMAT(/' DENSITY :',T25,5F10.3,/)
      WRITE(2,87) (ITERM(I),I=1,NSET)
   87 FORMAT(/,' TERMINATION MODE :',T25,5I10)
      WRITE(2,95)(LDSTR(I),I=1,NSET)
   95 FORMAT(/' DISTRIBUTION TYPE',//,'    FOR TRACE LENGTH :',
     *T25,5I10,/)
      WRITE(2,100)(THDSTR(I),I=1,NSET)
  100 FORMAT('    FOR ORIENTATION :',T25,5I10)
C
      WRITE(2,108) (ALE(I),I=1,NSET)
  108 FORMAT(/' DISTRIBUTION PARAMETERS',///,'  FOR TRACE LENGTH',//,10X,
     *'A :',T25,5F10.2)
      WRITE(2,110) (BLE(I), I=1,NSET)
  110 FORMAT(10X,'B :',T25,5F10.2)
```

```
      WRITE(2,112) (ATH(I),I=1,NSET)
  112 FORMAT(/,'  FOR ORIENTATION WITHIN SIMULATION PLANE',//,10X,
     *'A :',T25,5F10.2)
      WRITE(2,115) (BTH(I), I=1,NSET)
  115 FORMAT(10X,'B :',T25,5F10.2)
      WRITE(2,117) (ATHP(I),I=1,NSET)
  117 FORMAT(/,'  FOR ORIENTATION FROM SIMULATION PLANE',//,10X,
     *'A :',T25,5F10.2)
      WRITE(2,119) (BTHP(I),I=1,NSET)
  119 FORMAT(10X,'B :',T25,5F10.2)
C
C  GENERATE THE FRACTURE TRACES.
C
      CALL GENLIN(X1,Y1,X2,Y2,XC,YC,TRACE,DEGRE,DEGREP,
     *TH,THP,IB,XORD,YORD,INC1,CUT1,CUT2)
C
      IF(ISP-1) 130,125,126
C
C
C  SAMPLE THE LINE NETWORK FOR SPACING VALUES.
C
  125 CALL SPCNG1(TANTH,X1,Y1,X2,Y2)
      GO TO 130
  126 CALL SPCNG2(TANTH,X1,Y1,X2,Y2)
C
C
C  LOCATE ALL THE INTERSECTIONS.
C
  130 CALL INTERS(TANTH,X1,Y1,X2,Y2,INC1,INC2,XORD,YORD,IB)
C
C
C  FIND OUT THE EFFECTIVENESS OF EVERY INTERSECTION.
C
      CALL EFFINT(X1,INC1,INC2,EFF,XORD,YORD,IB)
C
      IF(PLTSIZ .LT. 0.0) GO TO 140
C
C
C  MAKE A PLOT OF THE FRACTURE NETWORK.
C
      CALL PLOTFR(PLTSIZ,TANTH,X1,Y1,X2,Y2,XORD,YORD)
C
C
C  DEFINE THE ELEMENTS.
C
  140 CALL ELTDEF(X1,Y1,X2,Y2,INC1,INC2,XORD,YORD,APER,CUT1,CUT2,
     *TH,THP,ELOR,ELORP,ELLEN,W,NOD1,NOD2,ISET)
C
      STOP
      END
```

```
C ***********************************************************************
C
C  THE SUBROUTINE GENLIN GENERATES THE LINE TRACES FOR ALL THE
C  FRACTURE SETS, ONE SET AFTER THE OTHER. GENLIN CALLS THE SUBROUTINES
C  REC@DR AND CIR2DR.
C
      SUBROUTINE GENLIN(X1,Y1,X2,Y2,XC,YC,TRACE,DEGRE,DEGREP,
     *TH,THP,IB,XORD,YORD,INC1,CUT1,CUT2)
C
      DOUBLE PRECISION SEED,PI,RAD
      INTEGER ESTNFR,THDSTR
      COMMON /AREA1A/ XMINO,YMAXO,XMAXO,YMINO
      COMMON /AREA1B/ XMINI,YMAXI,XMAXI,YMINI,IGEOM
      COMMON /AREA2/ NSET,MAXFRA,MAXINT,MAXELT,NUMINT,NUMNPT,NUMELT,
     *NOINT,IPRT
      COMMON /AREA3A/ SEED,PI,RAD,COEF
      COMMON /AREA4/ ATH(5),BTH(5),ATHP(5),BTHP(5),THDSTR(5),
     *NFRAC(5)
      COMMON /AREA5/ DENS(5),ALE(5),BLE(5),LDSTR(5),ITERM(5)
      COMMON /AREA7/ R(8),XB(8),YB(8),ISHAP(8),IBC(8)
      COMMON /AREA9/ DESLEN,SUMLE,ESTNFR
      COMMON /AREA15/ JSET,JFRA,JO,NBN,IBL(2)
      COMMON /AREA16/ LINT
C
      INTEGER IB(MAXINT),INC1(MAXINT,2)
      INTEGER CUT1(MAXFRA,NSET),CUT2(MAXFRA,NSET)
      DIMENSION TH(MAXFRA,NSET),THP(MAXFRA,NSET),
     *X1(MAXFRA,NSET),Y1(MAXFRA,NSET),X2(MAXFRA,NSET),Y2(MAXFRA,NSET)
      DIMENSION XC(MAXFRA),YC(MAXFRA),TRACE(MAXFRA),DEGRE(MAXFRA),
     *DEGREP(MAXFRA)
      DIMENSION XORD(MAXINT),YORD(MAXINT)
C
C
C COMPUTE SURFACE AREA OF MODEL
      IF(IGEOM .EQ. 1) THEN
        AREA=(XMAXI-XMINI)*(YMAXI-YMINI)
      ELSE
        RO=R(3)
        RI=R(6)
        AREA=PI*(RO-RI)*(RO+RI)/4.
      END IF
C
      LINT=0
C
      DO 220 JSET=1,NSET
C
C CALCULATE DESIRED TOTAL TRACE LENGTH (DESLEN) AND ESTIMATED NUMBER OF
C TRACES (ESTNFR) FOR THE SET.
      I=JSET
      A1=ATHP(I)*RAD
      DENS(I)=DENS(I)*SIN(A1)
      DESLEN = DENS(I) * AREA
      IF(LDSTR(I) .EQ. 2) GO TO 16
      AMEAN=ALE(I)
      GO TO 18
```

```
   16 AMEAN = EXP(ALE(I)+ BLE(I)*BLE(I)*0.5)
   18 ESTNFR = (DESLEN/AMEAN) * COEF
      IF(ESTNFR .LT. MAXFRA) GO TO 24
      WRITE(2,20) ESTNFR,I,MAXFRA
   20 FORMAT('1**** ERROR CONCERNING MAXFRA **********',///,
     *' ESTNFR = ',I4,' FOR SET ',I4,'. THIS IS MORE THAN THE MAXIMUM'
     */,'FOR WHICH MEMORY SPACE IS AVAILABLE, I.E. ',I4,' (MAXFRA)'
     *,//' DECREASE COEF OR INCREASE THE VALUE OF MAXFRA IN THE PROGRAM')
      STOP
   24 IF(IPRT .EQ. 1)GO TO 26
      WRITE(2,25) I
   25 FORMAT(////,10X,'FRACTURES  OF  SET',I4,//4X,'NUMBER',//,
     *' GNRTD ACTUAL', 4X,'LENGTH',5X,'ANGLE',6X,'X1',8X,'Y1',8X,'X2',
     *8X,'Y2',//)
C
C    GENERATE FRACTURE CENTERS.
C
   26  CALL GGUBS(SEED,ESTNFR,XC)
       CALL GGUBS(SEED,ESTNFR,YC)
C
C    GENERATE  FRACTURE TRACE LENGTHS .
C
       IF(LDSTR(I) .EQ. 2) GO TO 30
       DO 27 J=1,ESTNFR
   27  TRACE(J) = ALE(I)
       GO TO 45
   30  CALL GGNLG(SEED,ESTNFR,ALE(I),BLE(I),TRACE)
C
C    GENERATE FRACTURE ORIENTATIONS .
C
   45  IF(THDSTR(I) .EQ. 2) GO TO 55
       DO 52 J=1,ESTNFR
       THP(J,I) = A1
   52  DEGRE(J)= ATH(I)
       GO TO 60
   55  CALL GGNML(SEED,ESTNFR,DEGRE)
       CALL GGNML(SEED,ESTNFR,DEGREP)
       DO 58 J=1,ESTNFR
       THP(J,I)= (DEGREP(J)*BTHP(I) + ATHP(I))*RAD
   58  DEGRE(J)= DEGRE(J)*BTH(I) + ATH(I)
C
C    COMPUTE COORDINATES OF EXTREMITIES OF ALL FRACTURES IN SET I.
C
   60  DO 70  J = 1,ESTNFR
       THL=DEGRE(J)
C
C   TO AVOID SINGULARITY AND EVENTUAL DIVISION BY 0., MAKE SURE THL IS
C   NOT EXACTLY -90.0,0.0,90.0,180.0 OR 270.0.
         IF((THL .GE. 0.50 .AND. THL .LE. 179.50) .OR. (THL .LE. -0.50)
     *  .OR. (THL .GE. 180.50)) GO TO 62
         IF((THL .LE. 0.5 .AND. THL .GE. 0.0) .OR.
     *     (THL .LE. 180.5 .AND. THL .GE. 180.0)) THEN
           THL=0.5
         ELSE IF((THL .LE. 0.0 .AND. THL .GE. -0.5) .OR.
     *     (THL .LE. 180.0 .AND. THL .GE. 179.5)) THEN
           THL=-0.5
         END IF
```

```
      GO TO 63
   62 IF((THL .LE. 89.5 .AND. THL .GE. -89.5) .OR.
     *    (THL .GE. 90.5 .AND. THL .LE. 269.5)) GO TO 63
      IF((THL .GE. 89.5 .AND. THL .LE. 90.0) .OR.
     *    (THL .GE. 269.5 .AND. THL .LE. 270.0)) THEN
          THL=89.5
      ELSE IF((THL .GE. 90.0 .AND. THL .LE. 90.5) .OR.
     *        (THL .GE. -90.0 .AND. THL .LE. -89.5)) THEN
          THL=90.5
      END IF
   63 THL=THL*RAD
      XCJ = XC(J)*XMAXO + (1-XC(J))*XMINO
      YCJ = YC(J)*YMAXO + (1-YC(J))*YMINO
      TRL=TRACE(J)
      DY = TRL*SIN(THL) * 0.5
      DX = TRL*COS(THL) * 0.5
      X1L = XCJ + DX
      Y1L = YCJ + DY
      X2L = XCJ - DX
      Y2L = YCJ - DY
C
C   TO SIMPLIFY COMPUTATIONS, MAKE SURE Y2 IS LT Y1.
C
      IF(Y2L .LT. Y1L) GO TO 65
      X3= X1L
      Y3= Y1L
      X1L = X2L
      Y1L = Y2L
      X2L = X3
      Y2L = Y3
C
C ARE THE FRACTURES OF THIS SET ABUTTING ON FRACTURES OF OTHER SETS ?
 65       WRITE(2,555)I,ITERM(I)
 555      FORMAT(' BEFORE TERM: I, ITERM(I):',2I5)
      IF (ITERM(I) .EQ. 1) THEN
         CALL TERM(JSET,THL,X1L,Y1L,X2L,Y2L,TH,X1,Y1,X2,Y2)
      END IF
C
C   CONVERT THE VALUE OF THL FROM ANGLE TO TANGENT AND ENTER VALUES IN
C   CORRESPONDING ARRAY.
C
      TH(J,I)=TAN(THL)
      X1(J,I)=X1L
      Y1(J,I)=Y1L
      X2(J,I)=X2L
      Y2(J,I)=Y2L
   70 CONTINUE
C
C    CHECK FOR ANY OF THE EXTREMITIES GOING OUTSIDE THE MODEL BOUNDARIES
C    AND MAKE ADJUSTMENT FOR THE TRACE LENGTH AND FOR THE COORDINATES
C     OF THE CHOPPED OUT EXTREMITY.
C
      NBN=0
      SUMLE=0.
      J0=0
      IF(IGEOM .EQ. 2)GO TO 80
      CALL RECBDR(X1,Y1,X2,Y2,TRACE,TH,DEGRE,IB,XORD,YORD,INC1,
```

```
     *CUT1,CUT2)
      GO TO 90
   80 CALL CIRBDR(X1,Y1,X2,Y2,TRACE,TH,DEGRE,IB,XORD,YORD,INC1,
     *CUT1,CUT2)
C
C  PRINT SUMMARY OF FRACTURE SET I.
C
   90 IF(SUMLE .GT. DESLEN) GO TO 180
      WRITE(2,175) DESLEN,SUMLE,ESTNFR
  175 FORMAT(///,'1**** ERROR IN SUBROUTINE GENLIN ****',///,
     *' ESTIMATED NUMBER OF FRACTURES IS TOO SMALL',///,
     *' INCREASE THE VALUE OF COEF.',///,
     *10X,'DESLEN = ',F10.4,' SUMLE = ',F10.4,' ESTNFR = ',I4)
      STOP
  180 ACTLEN = SUMLE
      NFRAC(I) = JFRA
      WRITE(2,214) I, DENS(I),AMEAN,AREA,DESLEN,ACTLEN,ESTNFR,NFRAC(I),
     *JO,NBN
  214 FORMAT(////////,20X,'SUMMARY OF FRACTURE SET',I4,
     *       //,2X,'DENSITY OF FRACTURING ',F10.4,' L/L**2',/
     *         ,2X,'THEORETICAL MEAN TRACE LENGTH',F10.4,/
     *         ,2X,'SURFACE AREA OF THE MODEL',F10.4,/
     *         ,2X,'DESIRED TOTAL LENGTH FOR THE SET',F10.4,/
     *         ,2X,'ACTUAL TOTAL LENGTH FOR THE SET ',F10.4,/
     *         ,2X,'ESTIMATED NUMBER OF FRACTURES',I4,/
     *         ,2X,'ACTUAL NUMBER OF FRACTURES GENERATED ',I4,/
     *   ,2X,'NUMBER OF FRACTURES TOTALLY IN THE BUFFER MARGIN ',I4,/
     *    ,2X,'NUMBER OF INTERSECTIONS WITH A BOUNDARY',I4)
  220 CONTINUE
C
      RETURN
      END
```

TERM

```
      SUBROUTINE TERM(JSET,THL,X1L,Y1L,X2L,Y2L,TANTH,X1,Y1,X2,Y2)
C
C ***************************************************************
C
C THIS SUBROUTINE ENSURES THAT THE EXTREMITY 1 (HIGHER Y-VALUE) ALWAYS
C TERMINATES ON ANOTHER FRACTURE, IF THE FRACTURE DOES INTERSECT
C ANOTHER FRACTURE.
C
C ***************************************************************
C
      COMMON /AREA2/ NSET,MAXFRA,MAXINT,MAXELT,NUMINT,NUMNPT,NUMELT,
     *NOINT,IPRT
      COMMON /AREA4/ ATH(5),BTH(5),ATHP(5),BTHP(5),THDSTR(5),NFRAC(5)
      COMMON /AREA5/ DENS(5),ALE(5),BLE(5),LDSTR(5),ITERM(5)
      DIMENSION TANTH(MAXFRA,NSET),X1(MAXFRA,NSET),Y1(MAXFRA,NSET),
     *X2(MAXFRA,NSET),Y2(MAXFRA,NSET)
C
        WRITE(2,5)JSET,X1L,Y1L,X2L,Y2L
  5     FORMAT(' OLD: JSET, X1L,Y1L,X2L,Y2L:',I5,4F10.2)
      IF (JSET .EQ. 1) THEN
        WRITE(2,10)
 10     FORMAT(///,' **** WARNING ****',//,
     *' FRACTURE SET 1 CANNOT HAVE ABUTTING FRACTURES, ',//,
     *' I.E. ITERM(1)=1 IS NOT PERMITTED.',////)
            RETURN
        END IF
C FIND ALL INTERSECTIONS OF THE FRACTURE CONSIDERED WITH THE
C FRACTURES OF THE SETS ALREADY GENERATED. THIS SECTION IS SIMILAR
C TO A PORTION OF SUBROUTINE INTERS.
C
      TAN1=1/TAN(THL)
      YMAX=0.
      DO 110 I=1,JSET-1
         NFRACI=NFRAC(I)
        DO 100 J=1,NFRACI
          TAN2=1/TANTH(J,I)
C COMPUTE INTERSECTION ORDINATE YI AND CHECK FOR YI INSIDE THE RANGE
C FOR BOTH FRACTURES.
          IF ((TAN1-TAN2) .EQ. 0.0) TAN1=TAN2+0.000001
          YI=((Y1L*TAN1)-(Y1(J,I)*TAN2)-X1L+X1(J,I)) / (TAN1-TAN2)
          IF (YI .LT. Y2L .OR. YI .GT. Y1L) GO TO 100
          IF (YI .LT. Y2(J,I) .OR. YI .GT. Y1(J,I)) GO TO 100
C CHECK FOR LARGEST Y-VALUE OF INTERSECTION
         IF (YI .GE. YMAX) YMAX=YI
C
 100    CONTINUE
 110  CONTINUE
      IF(YMAX .LE. 0.) RETURN
C
C CUT EXTREMITY 1
      XMAX=(YMAX-Y1L)*TAN1 + X1L
      DY=Y1L-YMAX
      DX=X1L-XMAX
      Y1L=YMAX+0.00001
      X1L=XMAX+0.00001*TAN1
```

```
 TERM


C
C ATTACH IT TO EXTREMITY 2
      X2L=X2L-DX
      Y2L=Y2L-DY
        WRITE(2,555)X1L,Y1L,X2L,Y2L
 555    FORMAT(' NEW:',22X,4F10.2)
C
      RETURN
      END
```

```
C ***********************************************************************
C
C THE SUBROUTINE RECBDR, FOR RECTANGULAR BOUNDARIES, CHECKS FOR ANY LINE
C EXTREMITY GOING OUTSIDE THE MODEL BOUNDARIES AND MAKES ADJUSTMENT FOR
C TRACE LENGTH AND FOR THE COORDINATES OF THE CHOPPED OUT EXTREMITY(IES)
C RECBDR CALLS THE SUBROUTINES CHOPXY AND BNODE.
C
      SUBROUTINE RECBDR(X1,Y1,X2,Y2,TRACE,TH,DEGRE,IB,XORD,YORD,INC1,
     *CUT1,CUT2)
      INTEGER ESTNFR,CUTL(2)
      LOGICAL ELIM
      COMMON /AREA19/ XMINI,YMAXI,XMAXI,YMINI,IGEOM
      COMMON /AREA2/ NSET,MAXFRA,MAXINT,MAXELT,NUMINT,NUMNPT,NUMELT,
     *NOINT,IPRT
      COMMON /AREA7/ P(8),XB(8),YB(8),ISHAP(8),IBC(8)
      COMMON /AREA9/ DESLEN,SUMLE,ESTNFR
      COMMON /AREA12/ X1L,Y1L,X2L,Y2L,TRL,THL,CUTL
      COMMON /AREA15/ JSET,JFRA,JC,NBN,IBL(2)
      COMMON /AREA16/ LINT
C
      INTEGER IB(MAXINT),INC1(MAXINT,2)
      INTEGER CUT1(MAXFRA,NSET),CUT2(MAXFRA,NSET)
      DIMENSION X1(MAXFRA,NSET),Y1(MAXFRA,NSET),X2(MAXFRA,NSET),
     *Y2(MAXFRA,NSET),TH(MAXFRA,NSET)
      DIMENSION DEGRE(MAXFRA),TRACE(MAXFRA)
      DIMENSION XORD(MAXINT),YORD(MAXINT)
      EQUIVALENCE (LINT,L),(JSET,I),(JFRA,J)
C
      DO 170 JFRA=1,ESTNFR
C
      ELIM=.FALSE.
      CUTL(1)=0
      CUTL(2)=0
      X1L=X1(J,I)
      Y1L=Y1(J,I)
      X2L=X2(J,I)
      Y2L=Y2(J,I)
      TRL=TRACE(J)
      THL=TH(J,I)
      IF(Y2L .GE. YMAXI .OR. Y1L .LE. YMINI)GO TO 180
C
C CASES OF INTERSECTION WITH A Y-BOUNDARY
      IF(Y1L .LE. YMAXI)GO TO 30
C CASE OF INTERSECTION WITH UPPER BOUNDARY
      CALL CHOPXY(1,2,YMAXI)
C
      IF(ISHAP(4) .EQ. 0) THEN
        IBL(1)=3
      ELSE
        IF(X1L .LE. XB(4)) THEN
          IBL(1)=3
        ELSE
          IBL(1)=4
        END IF
      END IF
```

```
C
   80 IF(Y2L .GE. YMINI)GO TO 85
C CASE OF INTERSECTION WITH LOWER BOUNDARY
      CALL CHOPXY(2,2,YMINI)
      IF(ISHAP(8) .EQ. 0) THEN
        I8L(2)=7
      ELSE
        IF(X2L .LE. X8(8)) THEN
          I8L(2)=8
        ELSE
          I8L(2)=7
        END IF
      END IF
C
   85 IF(THL .LE. 0.0)GO TO 105
C
C CASE X2L .LT. X1L (I.E. THL IS POSITIVE)
      IF(X2L .GE. XMAXI .OR. X1L .LE. XMINI)GO TO 160
      IF(X1L .LE. XMAXI)GO TO 100
C CASE OF INTERSECTION WITH RIGHT BOUNDARY
      CALL CHOPXY(1,1,XMAXI)
      IF(ISHAP(6) .EQ. 0) THEN
        I8L(1)=5
      ELSE
        IF(Y1L .LE. Y8(6)) THEN
          I8L(1)=6
        ELSE
          I8L(1)=5
        END IF
      END IF
C
  100 IF(X2L .GE. XMINI)GO TO 165
C CASE OF INTERSECTION WITH LEFT BOUNDARY
      CALL CHOPXY(2,1,XMINI)
      IF(ISHAP(2) .EQ. 0) THEN
        I8L(2)=1
      ELSE
        IF(Y2L .LE. Y8(2)) THEN
          I8L(2)=1
        ELSE
          I8L(2)=2
        END IF
      END IF
C
      GO TO 165
C
C CASE X2L .GT. X1L (I.E. THL IS NEGATIVE)
  105 IF(X2L .LE. XMINI .OR. X1L .GE. XMAXI)GO TO 160
      IF(X1L .GE. XMINI)GO TO 115
C CASE OF INTERSECTION WITH LEFT BOUNDARY
      CALL CHOPXY(1,1,XMINI)
      IF(ISHAP(2) .EQ. 0) THEN
        I8L(1)=1
      ELSE
        IF(Y1L .LE. Y8(2)) THEN
          I8L(1)=1
        ELSE
```

```
          IBL(1)=2
        END IF
      END IF
C
  115 IF(X2L .LE. XMAXI)GO TO 165
C CASE OF INTERSECTION WITH RIGHT BOUNDARY
      CALL CHOPXY(2,1,XMAXI)
      IF(ISHAP(6) .EQ. 0) THEN
        IBL(2)=5
      ELSE
        IF(Y2L .LE. YB(L)) THEN
          IBL(2)=6
        ELSE
          IBL(2)=5
        END IF
      END IF
C
      GO TO 165
C
  160 ELIM=.TRUE.
      X1L=999.
      JJ=JJ+1
  165 IF(IPRT .EQ. 1)GO TO 167
      WRITE(2,166)J,TRL,DEGRE(J),X1L,Y1L,X2L,Y2L
  166 FORMAT(I12,2X,6F10.3)
  167 X1(J,I)=X1L
      IF(ELIM) GO TO 170
C
C REENTER VALUES IN CORRESPONDING ARRAYS.
      SUMLE=SUMLE+TRL
      Y1(J,I)=Y1L
      X2(J,I)=X2L
      Y2(J,I)=Y2L
      TRACE(J)=TRL
C
C DEFINE BOUNDARY NODE(S)
      DO 168 MEND=1,2
      IF(CUTL(MEND) .EQ. 1) THEN
        CALL BNODE(MEND,IB,XORD,YORD,INC1)
      END IF
  168 CONTINUE
C
      CUT1(J,I)=CUTL(1)
      CUT2(J,I)=CUTL(2)
      IF(SUMLE .GE. DESLEN)GO TO 200
  170 CONTINUE
  200 RETURN
      END
```

```
C*********************************************************************
C
C THE SUBROUTINE CIRBDR, FOR CIRCULAR BOUNDARIES, CHECKS FOR ANY LINE
C EXTREMITY GOING OUTSIDE THE MODEL BOUNDARIES AND MAKES ADJUSTMENT
C FOR TRACE LENGTH AND FOR THE COORDINATES OF THE CHOPPED CUT
C EXTREMITY(IES). CIRBDR CALLS THE SUBROUTINES CIRCL2, CHOPXY, CHOPC
C AND BNODE.
C
C                  LOCAL VARIABLES (IN CIRBDR AND CIRCL2)
C
C CIRCL1 : LOGICAL STATEMENT FUNCTION THAT TAKES VALUE .TRUE. IF A GIVEN
C          POINT IS OUTSIDE A GIVEN CIRCLE.
C INTC1 & INTC2 :
C          LOGICAL VARIABLES THAT TAKE VALUE .TRUE. IF A GIVEN LINE
C          INTERSECTS A GIVEN CIRCLE (TWO POSSIBLE INTERSECTIONS).
C OUTC1 & OUTC2 :
C          LOGICAL VARIABLES WHOSE VALUE IS ASSIGNED BY THE STATEMENT
C          FUNCTION CIRCL1, FOR EXTREMITIES 1 & 2 RESPECTIVELY OF A
C          GIVEN LINE.
C SEPAR :  LOGICAL VARIABLE THAT TAKES VALUE .TRUE. IF A LINE IS CUT IN
C          TWO SEGMENTS INSIDE THE MODEL ITSELF BY THE INNER CIRCLE.
C XCENT & YCENT :
C          COORDINATES OF THE CENTER OF THE CIRCLES FORMING A BOUNDARY.
C XC1, YC1 & XC2, YC2 :
C          COORDINATES OF THE INTERSECTION(S) OF A LINE AND A CIRCLE
C          (POSSIBILITY OF TWO INTERSECTIONS).
C
      SUBROUTINE CIRBDR(X1,Y1,X2,Y2,TRACE,TH,DEGRE,IB,XORD,YORD,INC1,
     *CUT1,CUT2)
      INTEGER ESTNFR,CUTL(2),CUTEMP
      LOGICAL OUTC1,OUTC2,INTC1,INTC2,CIRCL1,SEPAR,ELIM
      COMMON /AREA1B/ XMINI,YMAXI,XMAXI,YMINI,IGEOM
      COMMON /AREA2/ NSET,MAXFRA,MAXINT,MAXELT,NUMINT,NUMNPT,NUMELT,
     *NOINT,IPRT
      COMMON /AREA7/ R(8),XB(8),YB(8),ISHAP(8),IBC(8)
      COMMON /AREA9/ DESLEN,SUMLE,ESTNFR
      COMMON /AREA12/ X1L,Y1L,X2L,Y2L,TRL,THL,CUTL
      COMMON /AREA14/ XCENT,YCENT
      COMMON /AREA15/ JSET,JFRA,JJ,NBN,IBL(2)
      COMMON /AREA16/ LINT
C
      INTEGER IB(MAXINT),INC1(MAXINT,2)
      INTEGER CUT1(MAXFRA,NSET),CUT2(MAXFRA,NSET)
      DIMENSION X1(MAXFRA,NSET),Y1(MAXFRA,NSET),X2(MAXFRA,NSET),
     *Y2(MAXFRA,NSET),TH(MAXFRA,NSET)
      DIMENSION DEGRE(MAXFRA),TRACE(MAXFRA)
      DIMENSION XORD(MAXINT),YORD(MAXINT)
      EQUIVALENCE (LINT,L),(JSET,I),(JFRA,JJ)
C
C CIRCL1 IS A STATEMENT FUNCTION THAT FINDS OUT IF A POINT IS OUTSIDE
C A GIVEN CIRCLE OR NOT.
      CIRCL1(X4,Y4,R2)=((X4-XMINI)**2 + (Y4-YMINI)**2) .GE. R2
C
C INITIALIZE OR SET THE VALUE OF SOME VARIABLES.
      XCENT=XMINI
```

```
 CIRBDR

      YCENT=YMINI
      J=0
      JJ=0
      RI2=R(6)**2
      RO2=R(3)**2
C
   15 SEPAR=.FALSE.
      J=J+1
      JJ=JJ+1
      ELIM=.FALSE.
      CUTL(1)=0
      CUTL(2)=0
      X1L=X1(JJ,I)
      Y1L=Y1(JJ,I)
      X2L=X2(JJ,I)
      Y2L=Y2(JJ,I)
      TRL=TRACE(JJ)
      THL=TH(JJ,I)
C
C FIND OUT IF POINTS ARE OUTSIDE OUTER CIRCLE
      OUTC1=CIRCL1(X1L,Y1L,RO2)
      OUTC2=CIRCL1(X2L,Y2L,RO2)
C CASE BOTH POINTS ARE INSIDE OUTER CIRCLE
      IF(.NOT. OUTC1 .AND. .NOT. OUTC2)GO TO 30
C
C LOCATE INTERSECTION(S) WITH OUTER CIRCLE
      CALL CIRCL2(RO2,INTC1,INTC2,XC1,YC1,XC2,YC2)
      IF(OUTC1 .AND. OUTC2) THEN
C CASE BOTH POINTS ARE OUTSIDE OUTER CIRCLE
        IF(.NOT. INTC1 .AND. .NOT. INTC2) THEN
          GO TO 160
        ELSE
          CALL CHOPC(1,XC1,YC1)
          IBL(1)=3
          CALL CHOPC(2,XC2,YC2)
          IBL(2)=3
        END IF
      ELSE
C CASES ONE POINT IN AND ONE POINT OUT OUTSIDE CIRCLE
        IF(OUTC1) THEN
          CALL CHOPC(1,XC1,YC1)
          IBL(1)=3
        ELSE
          CALL CHOPC(2,XC2,YC2)
          IBL(2)=3
        END IF
      END IF
C
C CHECK FOR INTERSECTION WITH LOWER BOUNDARY
C
C CASE OF LINE TOTALLY IN BOTTOM MARGIN
   30 IF(Y1L .LE. YMINI)GO TO 160
      IF(Y2L .LE. YMINI) THEN
C CASE OF LINE OVERLAPPING LOWER BOUNDARY
        CALL CHOPXY(2,2,YMINI)
        IF(ISHAP(5) .EQ. 0) THEN
          IBL(2)=4
```

```
          ELSE
            IF(X2L .LE. XB(5)) THEN
              IBL(2)=5
            ELSE
              IBL(2)=4
            END IF
          END IF
        END IF
C
C CHECK FOR INTERSECTION WITH LEFT BOUNDARY
C
C CASE OF LINE TOTALLY IN LEFT MARGIN
      IF(X1L .LE. XMINI .AND. X2L .LE. XMINI)GO TO 160
C CASE OF LINE TOTALLY TO THE RIGHT OF LEFT BOUNDARY
      IF(X1L .GE. XMINI .AND. X2L .GE. XMINI)GO TO 40
C CASES OF LINE OVERLAPPING LEFT BOUNDARY
      IF(X1L .LE. XMINI) THEN
        CALL CHOPXY(1,1,XMINI)
        IF(ISHAP(2) .EQ. 0) THEN
          IBL(1)=1
        ELSE
          IF(Y1L .LE. YB(2)) THEN
            IBL(1)=1
          ELSE
            IBL(1)=2
          END IF
        END IF
      ELSE
        CALL CHOPXY(2,1,XMINI)
        IF(ISHAP(2) .EQ. 0) THEN
          IBL(2)=1
        ELSE
          IF(Y2L .LE. YB(2)) THEN
            IBL(2)=1
          ELSE
            IBL(2)=2
          END IF
        END IF
      END IF
C
C FIND OUT IF POINTS ARE OUTSIDE INNER CIRCLE.
C (N.B. LINES ARE NOW LIMITED TO FIRST QUADRANT)
   40 OUTC1=CIRCL1(X1L,Y1L,RI2)
      OUTC2=CIRCL1(X2L,Y2L,RI2)
C CASES BOTH POINTS INSIDE OR BOTH POINTS OUTSIDE INNER CIRCLE
      IF(.NOT. OUTC1 .AND. .NOT. OUTC2) THEN
        GO TO 160
      ELSE
        IF(OUTC1 .AND. OUTC2 .AND. THL .GE. 0.0)GO TO 165
      END IF
C
C LOCATE INTERSECTION WITH INNER CIRCLE
      CALL CIRCL2(RI2,INTC1,INTC2,XC1,YC1,XC2,YC2)
      IF(OUTC1 .AND. OUTC2) THEN
C CASE BOTH POINTS OUTSIDE INNER CIRCLE
        IF(.NOT. INTC1) THEN
          GO TO 165
```

```
          ELSE
            GO TO 50
          END IF
        END IF
C CASES ONE POINT IN AND THE OTHER OUT INNER CIRCLE
        IF(OUTC1) THEN
          CALL CHOPC(2,XC1,YC1)
          IBL(2)=6
        ELSE
          CALL CHOPC(1,XC2,YC2)
          IBL(1)=6
        END IF
C
        GO TO 165
C
C CASE OF BOTH POINTS OUTSIDE INNER CIRCLE AND EXISTENCE OF INTERSECTION
C WITH THIS CIRCLE (I.E. LINE SEPARATED IN TWO SEGMENTS).
   50 SEPAR=.TRUE.
          XTEMP=X2L
          YTEMP=Y2L
          CUTEMP=CUTL(2)
          IBTEMP=IBL(2)
        X2L=XC1
        Y2L=YC1
        CUTL(2)    =1
        IBL(2)=6
        GO TO 60
C
   55 CONTINUE
C (N.B. THE ENTRIES ORIGINALLY IN LOCATION JJ ARE WRITTEN OVER BY THE
C  ENTRIES RELATED TO THE SECOND SEGMENT OF LINE J)
        SEPAR=.FALSE.
        JJ=JJ+1
        X1L=XC2
        Y1L=YC2
        X2L=XTEMP
        Y2L=YTEMP
        CUTL(2)=CUTEMP
        IBL(2)=IBTEMP
        CUTL(1)    =1
        IBL(1)=6
C
   60 TRL=SQRT((X1L-X2L)**2 + (Y1L-Y2L)**2)
        GO TO 165
C
  160 ELIM=.TRUE.
        X1L=999.
        JO=JO+1
  165 IF(IPRT .EQ. 1)GO TO 170
        WRITE(2,166) J,JFRA,TRL,DEGRE(J),X1L,Y1L,X2L,Y2L
  166 FORMAT(2I6,2X,6F10.3)
  170 X1(JJ,I)=X1L
        IF(ELIM) GO TO 190
C
C REENTER VALUES IN CORRESPONDING ARRAYS.
        SUMLE=SUMLE+TRL
        Y1(JJ,I)=Y1L
```

```
CIRBDR

      X2(JJ,I)=X2L
      Y2(JJ,I)=Y2L
      TRACE(JJ)=TRL
      TH(JJ,I)=THL
C
C DEFINE BOUNDARY NODES.
      DO 180 MEND=1,2
      IF(CUTL(MEND) .EQ. 1) THEN
        CALL BNODE(MEND,IB,XORD,YORD,INC1)
      END IF
  180 CONTINUE
C
      CUT1(JJ,I)=CUTL(1)
      CUT2(JJ,I)=CUTL(2)
      IF(SUMLE .GE. DESLEN)GO TO 200
      IF(SEPAR)GO TO 55
  190 IF(JJ .LT. ESTNFR)GO TO 15
C
  200 RETURN
      END
```

CIRCL2

```
C
C*************************************************************************
C
C  THE SUBROUTINE CIRCL2, USED FOR CIRCULAR BOUNDARIES, CHECKS IF
C  INTERSECTION(S) OF A LINE AND A CIRCLE EXIST WITHIN THE RANGE OF THE
C  FINITE LINE, AND CALCULATES THE COORDINATES OF THIS (THESE)
C  INTERSECTION(S). CIRCL2 FINDS THE SOLUTIONS OF A QUADRATIC EQUATION O
C  THE FORM : A*X**2 + B*X + C = ).
C
C                          LOCAL VARIABLES
C
C  B & C :   VALUE OF B & C IN THE QUADRATIC EQUATION
C  DISCR :   VALUE OF THE DISCRIMINANT, I.E. B**2-4*A*C, IN THE SOLUTION
C  FOURAC :  VALUE OF THE EXPRESSION 4*A*C.
C            OF THE QUADRATIC EQUATION.
C  OTAN :    VALUE OF 1/TH(J,I) FOR FRACTURE J IN SET I.
C  TWOA :    VALUE OF THE EXPRESSION 2*A
C  YOTAN :   VALUE OF Y1(J,I)/TH(J,I) FOR FRACTURE J IN SET I.
C
      SUBROUTINE CIRCL2(R2,INTC1,INTC2,X1,Y1,X2,Y2)
C
      INTEGER CUTL(2)
      LOGICAL INTC1,INTC2
      COMMON /AREA12/ X1L,Y1L,X2L,Y2L,TRL,THL,CUTL
      COMMON /AREA14/ XCENT,YCENT
C
      INTC1=.FALSE.
      INTC2=.FALSE.
      OTAN=1/THL
      TWOA=2+2*OTAN*OTAN
      YOTAN=Y1L*OTAN
      B=2*OTAN*(X1L-YOTAN-XCENT)-2*YCENT
      B2=B*B
      C=2*YOTAN*(XCENT-X1L)-2*X1L*XCENT+X1L*X1L+XCENT*XCENT
     **+YCENT*YCENT+YOTAN*YOTAN-R2
      FOURAC=2*TWOA*C
      DISCR=B2-FOURAC
      IF(DISCR .LE. 0.0)GO TO 200
      DISCR=SQRT(DISCR)
C
C COMPUTE THE COORDINATES OF THE FIRST INTERSECTION AND CHECK IF WITHIN
C THE RANGE OF THE FINITE LINE.
      Y1=(-B+DISCR)/TWOA
      IF(Y1 .GE. Y1L .OR. Y1 .LE. Y2L)GO TO 50
      X1=X1L+OTAN*(Y1-Y1L)
      INTC1=.TRUE.
C
C DO THE SAME FOR SECOND INTERSECTION
   50 Y2=(-B-DISCR)/TWOA
      IF(Y2 .GE. Y1L .OR. Y2 .LE. Y2L)GO TO 200
      X2=X1L+OTAN*(Y2-Y1L)
      INTC2=.TRUE.
C
  200 RETURN
      END
```

BNODE

```
C*********************************************************************
C
C THE SUBROUTINE BNODE IS CALLED BY THE SUBROUTINES RECBDR OR CIRBDR
C TO SET THE VALUE OF NBN AND OF THE INTERSECTION NUMBER (L), AND TO
C RECORD THE COORDINATES AND INCIDENCE OF INTERSECTION L.
C
      SUBROUTINE BNODE(MEND,IB,XORD,YORD,INC1)
C
      INTEGER CUTL(2)
      COMMON /AREA2/ NSET,MAXFRA,MAXINT,MAXELT,NUMINT,NUMNPT,NUMELT,
     *NOINT,IPRT
      COMMON /AREA7/ R(8),XB(8),YB(8),ISHAP(8),IBC(8)
      COMMON /AREA12/ X1L,Y1L,X2L,Y2L,TRL,THL,CUTL
      COMMON /AREA15/ JSET,JFRA,JO,NBN,IBL(2)
      COMMON /AREA16/ LINT
      DIMENSION XORD(MAXINT),YORD(MAXINT)
      INTEGER INC1(MAXINT,2),IB(MAXINT)
      EQUIVALENCE (LINT,L)
C
      IF(IBC(IBL(MEND)) .GT. 0) CUTL(MEND)=2
C
      L=L+1
      NBN=NBN+1
      IF(MEND .EQ. 1) THEN
        XL=X1L
        YL=Y1L
      ELSE
        XL=X2L
        YL=Y2L
      END IF
      XORD(L)=XL
      YORD(L)=YL
      INC1(L,1)=JSET
      INC1(L,2)=JFRA
      IB(L)=IBL(MEND)
C
      RETURN
      END
```

CHOPC

```
C
C*********************************************************************
C
C  THE SUBROUTINE CHOPC IS CALLED BY THE SUBROUTINE CIRBDR TO COMPUTE
C  THE CORRECTED LENGTH AND THE NEW COORDINATES OF THE CHOPPED OUT
C  EXTREMITY OF A LINE GOING ACROSS A CIRCULAR BOUNDARY.
C
      SUBROUTINE CHOPC(MEND,XI,YI)
C
      INTEGER CUTL(2)
      COMMON /AREA12/ X1L,Y1L,X2L,Y2L,TRL,THL,CUTL
C
      CUTL(MEND)=1
      IF(MEND .EQ. 1) THEN
        X=X1L
        Y=Y1L
      ELSE
        X=X2L
        Y=Y2L
      END IF
      DIFLE=SQRT((X-XI)**2+(Y-YI)**2)
      TRL=TRL-DIFLE
C
      IF(MEND .EQ. 1) THEN
        X1L=XI
        Y1L=YI
      ELSE
        X2L=XI
        Y2L=YI
      END IF
C
      RETURN
      END
```

```
C*********************************************************************
C
C THE SUBROUTINE CHOPXY IS CALLED BY THE SUBROUTINES RECBDR OR CIRBDR
C TO COMPUTE THE CORRECTED LENGTH AND THE NEW COORDINATES OF THE
C CHOPPED-OUT EXTREMITY OF A LINE GOING ACROSS A MODEL BOUNDARY.
C
C                        LOCAL VARIABLES
C
C IXY :     KEY DESIGNING WHICH AXIS THE CONSIDERED BOUNDARY IS CROSSING
C           1, FOR A BOUNDARY CROSSING THE X-AXIS,
C           2, FOR A BOUNDARY CROSSING THE Y-AXIS.
C
      SUBROUTINE CHOPXY(MEND,IXY,EXTR)
C
      INTEGER CUTL(2)
      COMMON /AREA12/ X1L,Y1L,X2L,Y2L,TRL,THL,CUTL
C
      CUTL(MEND)=1
      IF(MEND .EQ. 1) THEN
        X=X1L
        Y=Y1L
      ELSE
        X=X2L
        Y=Y2L
      END IF
      IF(IXY .EQ. 1) THEN
        DIFX=X-EXTR
        X=EXTR
        DIFY=DIFX*THL
        Y=Y-DIFY
      ELSE
        DIFY=Y-EXTR
        Y=EXTR
        DIFX=DIFY/THL
        X=X-DIFX
      END IF
      TRL=TRL-SQRT(DIFX*DIFX+DIFY*DIFY)
C
      IF(MEND .EQ. 1) THEN
        X1L=X
        Y1L=Y
      ELSE
        X2L=X
        Y2L=Y
      END IF
      RETURN
      END
```

```
C ******************************************************************
C
C                    LOCAL VARIABLES (FOR SPCNG1 & SPCNG2)
C
C CORR : CORRECTION FACTOR ACCOUNTING FOR RELATIVE ORIENTATION OF MODEL
C        PLANE (ALSO SAMPLING LINE IN SPCNG2) AND FRACTURE SET BEING
C        SAMPLED.
C IL : COUNTING INDEX FOR SAMPLING LINE.
C INT : INDEX FOR COUNTING THE INTERSECTIONS ON THE CURRENT SAMPLING
C       LINE WITH THE CURRENT FRACTURE SET.
C IR(INT) : PERMUTATION MADE ON YINT(INT) BY SUBROUTINE VSRTR.
C JFRAC(INT) : FRACTURE NUMBER IN THE SET BEING SAMPLED, CORRESPONDING
C              TO THE INTERSECTION INT ON THE CURRENT SAMPLING LINE.
C MAXIL : MAXIMUM NUMBER OF INTERSECTIONS OF A SAMPLING LINE WITH
C         FRACTURES OF THE SAME SET.
C NSP : NUMBER OF SPACING VALUES GENERATED THUS FAR FOR THE FRACTURE
C       SET BEING SAMPLED ( NSP(I) FOR SET I IN SPCNG2).
C NTEN : NUMBER OF TIMES 10 SAMPLING LINES HAVE BEEN GENERATED ( FOR
C        THE CURRENT FRACTURE SET IN SPCNG1 ).
C SPAC(NSP) : SPACING VALUE.
C TANLIN : TANGENT OF THE ANGLE OF THE CURRENT SAMPLING LINE.
C XINT(INT) & YINT(INT) : COORDINATES OF INTERSECTION INT ON THE
C           CURRENT SAMPLING LINE ( WITH THE CURRENT FRACTURE SET IN
C           SPCNG2)
C XP(IL) & YP(IL) : COORDINATES OF THE POINT USED TO GENERATE THE
C                   SAMPLING LINE IL.
C
C
C ******************************************************************
C
C THE SUBROUTINE SPCNG1 COMPUTE 150 SPACING VALUES FOR EACH FRACTURE
C SET. THE SAMPLING LINES ARE WITHIN THE SIMULATION PLANE. THEY ARE
C PERPENDICULAR TO THE AVERAGE ORIENTATION OF THE LINES OF INTERSECTION
C OF THE FRACTURES OF THE SET BEING SAMPLED.
C
      SUBROUTINE SPCNG1(TANTH,X1,Y1,X2,Y2)
C
      DOUBLE PRECISION SEED,PI,RAD
      INTEGER THDSTR
      COMMON /AREA1B/ XMINI,YMAXI,XMAXI,YMINI,IGEOM
      COMMON /AREA2/ NSET,MAXFRA,MAXINT,MAXELT,NUMINT,NUMNPT,NUMELT,
     *NOINT,IPRT
      COMMON /AREA3A/ SEED,PI,RAD,COEF
      COMMON /AREA3B/ TITLE(20)
      COMMON /AREA4/ ATH(5),BTH(5),ATHP(5),BTHP(5),THDSTR(5),NFRAC(5)
      DIMENSION TANTH(MAXFRA,NSET),X1(MAXFRA,NSET),Y1(MAXFRA,NSET),
     *X2(MAXFRA,NSET),Y2(MAXFRA,NSET)
      DIMENSION XP(10),YP(10),XINT(70),YINT(70),JFRAC(70),IR(70)
      DATA MAXIL/70/
C
      WRITE(3,10) TITLE
10    FORMAT('1',//,1X,20A4,//,' FILE: SPACING1.U03')
      DO 250 I=1,NSET
      A=ATHP(I) * RAD
      CORR=SIN(A)
```

```
      THSL = ATH(I) + 90.0
C
C MAKE SURE THAT SAMPLING LINE IS NOT SUB-PARALLEL TO X-AXIS
      IF((THSL .GE. 0.01 .AND. THSL .LE. 179.99) .OR. (THSL .LE. -0.01)
     *.OR. (THSL .GE. 180.01))GO TO 15
      THSL=0.01
   15 THSL=THSL*RAD
      TANLIN = 1 / TAN(THSL)
      NFRA = NFRAC(I)
      NSP = 1
      NTEN = 1
      WRITE(3,20) I
   20 FORMAT(/////,' FRACTURE SET ',I4,///,' NUMB',' NTEN','  LINE',
     *4X,'XP',6X,'YP',3X,'FRACT','  SPACING',//)
C
C GENERATE 10 RANDOM POINTS IN THE MODEL.
C
   30 CALL GGUBS(SEED,10,XP)
      CALL GGUBS(SEED,10,YP)
      DO 190 IL=1,10
      XP(IL) = XP(IL)*XMAXI + (1-XP(IL))*XMINI
      YP(IL) = YP(IL)*YMAXI + (1-YP(IL))*YMINI
      INT = 1
      DO 150 J=1,NFRA
      IF(X1(J,I) .GE. 999.0) GO TO 150
      TANFR = 1 / TANTH(J,I)
C
C COMPUTE INTERSECTION ORDINATE YI, AND CHECK FOR YI INSIDE THE RANGE
C FOR THE FRACTURE (J,I).
C
      YI = ((Y1(J,I)*TANFR) - (YP(IL)*TANLIN) - X1(J,I) + XP(IL) )
     * / (TANFR - TANLIN)
      IF(YI .LT. Y2(J,I) .OR. YI .GT. Y1(J,I) ) GO TO 150
C
C COMPUTE INTERSECTION ABSCISSA AND PUT THE COORDINATES IN XINT(40)
C & YINT(40) ARRAYS.
C
      XI = (YI-Y1(J,I)) * TANFR  +  X1(J,I)
      XINT(INT) = XI
      YINT(INT) = YI
      JFRAC(INT) = J
      IR(INT)=INT
      INT = INT+1
      IF(INT .LE. MAXIL) GO TO 150
      WRITE(2,145) INT,NTEN,IL,I
  145 FORMAT(/////,' **** ERROR IN SUBROUTINE SPCNG1****',//,
     *' THERE ARE AT LEAST ',I4,' INTERSECTIONS ON SAMPLING LINE ',
     *I4,' *10 +',I4,' OF SET ',I4,//,' INCREASE THE VALUE OF MAXIL '
     *,'AND THE DIMENSION OF XINT, YINT, JFRAC AND IR')
      STOP
  150 CONTINUE
      INT = INT-1
      IF(INT .LE. 1) GO TO 190
C
C ORDER THE INTERSECTIONS BY INCREASING Y VALUES.
C
      CALL VSRTP(YINT,INT,IR)
```

```
SPCNG1

          DO 160 M1=1,INT
          JTEMP=JFRAC(M1)
          JFRAC(M1)=JFRAC(IR(M1))
  160     JFRAC(IR(M1))=JTEMP
          DO 165 M1=1,INT
          XTEMP=XINT(M1)
          XINT(M1)=XINT(IR(M1))
  165     XINT(IR(M1))=XTEMP
C
C CALCULATE SPACING BETWEEN CONSECUTIVE INTERSECTIONS.
C
      DO 175 ISP=2,INT
      DX = XINT(ISP) - XINT(ISP-1)
      DY = YINT(ISP) - YINT(ISP-1)
      SPAC = SQRT(DX*DX + DY*DY) * CORR
      WRITE(3,170) NSP,NTEN,IL,XP(IL),YP(IL),JFRAC(ISP),SPAC
  170 FORMAT(3I5,2F8.2,I5,F10.2)
      NSP = NSP+1
  175 CONTINUE
      IF(NSP-151) 190,250,250
  190 CONTINUE
      IF(NSP-151) 195,250,250
  195 NTEN = NTEN+1
      GO TO 30
  250 CONTINUE
      RETURN
      END
```

```
C ***************************************************************************
C
C THE SUBROUTINE SPCNG2 COMPUTES 150 SPACING VALUES FOR EACH FRACTURE
C SET. THE SAMPLING LINES ARE RANDOMLY ORIENTED WITHIN THE SIMULATION
C PLANE.
C
      SUBROUTINE SPCNG2(TANTH,X1,Y1,X2,Y2)
C
      DOUBLE PRECISION SEED,PI,RAD
      INTEGER THDSTR
      COMMON /AREA1B/ XMINI,YMAXI,XMAXI,YMINI,IGEOM
      COMMON /AREA2/ NSET,MAXFRA,MAXINT,MAXELT,NUMINT,NUMNPT,NUMELT,
     *NOINT,IPRT
      COMMON /AREA3A/ SEED,PI,RAD,COEF
      COMMON /AREA3B/ TITLE(20)
      COMMON /AREA4/ ATH(5),BTH(5),ATHP(5),BTHP(5),THDSTR(5),NFRAC(5)
      DIMENSION TANTH(MAXFRA,NSET),X1(MAXFRA,NSET),Y1(MAXFRA,NSET),
     *X2(MAXFRA,NSET),Y2(MAXFRA,NSET)
      DIMENSION XP(10),YP(10),THSL(10),NSP(5)
      DIMENSION XINT(70),YINT(70),JFRAC(70),IR(70)
      DATA MAXIL/70/,NSP/5*1/
C
      WRITE(4,10) TITLE
10    FORMAT('1',//,1X,20A4,//,' FILE: SPACING2.U04')
      NTEN = 1
C
      WRITE(4,20)
   20 FORMAT(/////,'  NTEN',' LINE',4X,'XP',6X,'YP',' ORIENTATION ',
     *'  SET',3X,'ANGLE',3X,'NUMB ','FRACT','  SPACING',//)
C
C GENERATE 10 RANDOM POINTS AND 10 RANDOM ORIENTATIONS.
C
   30 CALL GSUBS(SEED,10,XP)
      CALL GSUBS(SEED,10,YP)
      CALL GSUBS(SEED,10,THSL)
C
C
      DO 300 IL=1,10
      XPIL = XP(IL)*XMAXI + (1-XP(IL))*XMINI
      YPIL = YP(IL)*YMAXI + (1-YP(IL))*YMINI
      TH=THSL(IL)*180.
C
C MAKE SURE THAT SAMPLING LINE IS NOT SUB-PARALLEL TO X-AXIS
      IF((TH .GE. 0.01 .AND. TH .LE. 179.99) .OR. (TH .LE. -0.01)
     *.OR. (TH .GE. 180.01)) GO TO 35
      TH=0.01
   35 TH=TH*RAD
      TANLIN=1/TAN(TH)
C
C
      DO 250 I=1,NSET
        IF(NSP(I) .GE. 150)GO TO 250
      INT = 1
      NFRA = NFRAC(I)
      ANGLE = ABS(THSL(IL)-ATH(I))
```

```
      SPCNG2


            A1=ANGLE*RAD
            A2=ATHP(I) * RAD
            CORR = SIN(A1) * SIN(A2)
            DO 150 J=1,NFRA
            IF(X1(J,I) .GE. 999.0) GO TO 150
            TANFR = 1 / TANTH(J,I)
            XX1=X1(J,I)
            YY1=Y1(J,I)
      C
      C COMPUTE INTERSECTION ORDINATE YI, AND CHECK FOR YI INSIDE THE RANGE
      C FOR THE FRACTURE (J,I).
      C
            YI = ((YY1     *TANFR) - (YPIL   *TANLIN) - XX1      + XPIL    )
           * / (TANFR - TANLIN)
            IF(YI .LT. Y2(J,I) .OR. YI .GT. YY1      ) GO TO 150
      C
      C COMPUTE INTERSECTION ABSCISSA AND PUT THE COORDINATES IN XINT(40)
      C & YINT(40) ARRAYS.
      C
            XI = (YI-YY1     ) * TANFR  +  XX1
            XINT(INT) = XI
            YINT(INT) = YI
            JFRAC(INT) = J
            IR(INT)=INT
            INT = INT+1
            IF(INT .LE. MAXIL) GO TO 150
            WRITE(2,145) INT,I,NTEN,IL
       145 FORMAT(/////,' **** ERROR IN SUBROUTINE SPCNG2****',//,
           *' THERE ARE AT LEAST ',I4,' INTERSECTIONS WITH FRACTURES OF SET '
           *,I4,' ON SAMPLING LINE ',I4,' *10 +',I4,//,' INCREASE THE VALUE OF
           *MAXIL AND THE DIMENSION OF XINT, YINT, JFRAC AND IR')
            STOP
       150 CONTINUE
            INT = INT-1
            IF(INT .LE. 1) GO TO 250
      C
      C ORDER THE INTERSECTIONS BY INCREASING Y VALUES.
      C
           CALL VSRTR(YINT,INT,IR)
              DO 160 M1=1,INT
              JTEMP=JFRAC(M1)
              JFRAC(M1)=JFRAC(IR(M1))
       160    JFRAC(IR(M1))=JTEMP
              DO 165 M1=1,INT
              XTEMP=XINT(M1)
              XINT(M1)=XINT(IR(M1))
       165    XINT(IR(M1))=XTEMP
      C
      C CALCULATE SPACING BETWEEN CONSECUTIVE INTERSECTIONS.
      C
           DO 175 ISP=2,INT
           DX = XINT(ISP) - XINT(ISP-1)
           DY = YINT(ISP) - YINT(ISP-1)
           SPAC = CORR * SQRT(DX*DX + DY*DY)
          WRITE(4,170) NTEN,IL,XPIL,YPIL,THSL(IL),I,ANGLE,NSP(I),
         *JFRAC(ISP),SPAC
       170 FORMAT(2I5,2F8.2,F8.1,3X,I5,3X,F5.1,2X,I5,I5,F10.2)
```

```
  175 NSP(I) = NSP(I)+1
  250 CONTINUE
C
C CHECK IF ALL NSP(I) ARE .GE. 150
      DO 260 I=1,NSET
  260 IF(NSP(I) .LT. 150)GO TO 300
      GO TO 350
  300 CONTINUE
      NTEN = NTEN+1
      GO TO 30
  350 RETURN
      END
```

```
C ***********************************************************************
C
C THE SUBROUTINE INTERS FINDS ALL THE INTERSECTIONS (OR NODES) BETWEEN
C THE LINE SEGMENTS, COMPUTE THE COORDINATES OF THESE NODES AND RECORD
C THEIR INCIDENCE.
C
      SUBROUTINE INTERS(TANTH,X1,Y1,X2,Y2,INC1,INC2,XORD,YORD,IB)
C
      INTEGER THDSTR
      COMMON /AREA1B/ XMINI,YMAXI,XMAXI,YMINI,IGEOM
      COMMON /AREA2/ NSET,MAXFRA,MAXINT,MAXELT,NUMINT,NUMNPT,NUMELT,
     *NOINT,IPRT
      COMMON /AREA4/ ATH(5),BTH(5),ATHP(5),BTHP(5),THDSTR(5)
     *,NFRAC(5)
      COMMON /AREA16/ LINT
      DIMENSION TANTH(MAXFRA,NSET),X1(MAXFRA,NSET),Y1(MAXFRA,NSET),
     *X2(MAXFRA,NSET),Y2(MAXFRA,NSET)
      DIMENSION XORD(MAXINT),YORD(MAXINT)
      INTEGER INC1(MAXINT,2),INC2(MAXINT,2),IB(MAXINT)
      EQUIVALENCE (LINT,L)
C
C N.B. BOUNDARY NODES HAVE ALREADY BEEN DEFINED IN SUBROUTINE GENLIN AND
C      THE OTHER SUBROUTINES CALLED BY GENLIN.
C
      DO 200 I=1,NSET
        NFRAC1 = NFRAC(I)
        IF(THDSTR(I) .EQ. 2) GO TO 8
          IBEG = I+1
          IF(IBEG .GT. NSET)GO TO 200
          GO TO 9
    8   IBEG = I
    9 DO 195 J=1,NFRAC1
          IF(X1(J,I) .GE. 999.0) GO TO 195
          X1L=X1(J,I)
          Y1L=Y1(J,I)
          X2L=X2(J,I)
          Y2L=Y2(J,I)
          TAN1=1/TANTH(J,I)
C
C  CHECK ALL THE LINES IN THE NETWORK FOR ANY POSSIBLE INTERSECTION WITH
C  LINE (J,I).
C
      DO 188 II=IBEG,NSET
        NFRAC2 = NFRAC(II)
        IF(II .GT. I) GO TO 24
        JBEG = J+1
        IF(JBEG .LE. NFRAC2) GO TO 25
        GO TO 188
   24   JBEG = 1
   25   DO 185 JJ=JBEG,NFRAC2
        IF(X1(JJ,II) .GE. 999.0) GO TO 185
        TAN2 = 1 / TANTH(JJ,II)
        IF(ABS(TAN1-TAN2) .LE. 1DE-6)GO TO 185
C
C  COMPUTE INTERSECTION ORDINATE (YI) , AND CHECK FOR YI INSIDE THE
```

```
INTERS

C  RANGE FOR BOTH FRACTURES.
C
        YI = ( (Y1L*TAN1)-(Y1(JJ,II)*TAN2)-X1L+X1(JJ,II) )
     *  / (TAN1 - TAN2)
        IF(YI .LT. Y2L .OR. YI .GT. Y1L) GO TO 185
        IF(YI .LT. Y2(JJ,II) .OR. YI .GT. Y1(JJ,II)) GO TO 185
C
C  COMPUTE INTERSECTION ABSCISSA (XI).
C
        XI = (YI-Y1L) * TAN1    + X1L
C
C  PUT THE INTERSECTION COORDINATES INTO THE XORD & YORD ARRAYS,
C  AND RECORD THE INCIDENCE OF THIS INTERSECTION.
C
        L=L+1
        XORD(L) = XI
        YORD(L) = YI
        INC1(L,1) = I
        INC1(L,2) = J
        INC2(L,1) = II
        INC2(L,2) = JJ
  185 CONTINUE
  188 CONTINUE
  195 CONTINUE
  200 CONTINUE
      NUMINT = L
C
      IF(NUMINT .LT. MAXINT) GO TO 210
      WRITE(2,206) NUMINT,MAXINT
  206 FORMAT('1**** ERROR CONCERNING MAXINT ****',///,
     *' THERE ARE ',I4,' INTERSECTIONS  AND THERE IS MEMORY SPACE',
     */,'AVAILABLE FOR ',I4,' (MAXINT)')
      STOP
  210 RETURN
      END
```

```
EFFINT


C ***********************************************************************
C
C THE SUBROUTINE EFFINT DETERMINES THE FLOW-EFFECTIVENESS OF EVERY
C INTERSECTION AND RECORD THE APPROPRIATE VALUE (I.E. 0 OR 1)
C IN THE ARRAY NOEFF.
C
C                     LOCAL VARIABLES
C
C II : NUMBER OF ITERATIONS OF THE COMPLETE LOOP OVER ALL THE FRACTURES
C INTER : NUMBER OF THE LAST EFFECTIVE INTERSECTION FOUND ON A FRACTURE
C NOINT : SUM OF ALL THE NON-EFFECTIVE INTERSECTIONS AFTER THE PREVIOUS
C         ITERATION.
C NOINT2 : SUM OF ALL THE NON-EFFECTIVE INTERSECTIONS AT THE END OF
C          CURRENT ITERATION.
C
      SUBROUTINE EFFINT(X1,INC1,INC2,EFF,XORD,YORD,IB)
      INTEGER THDSTR
      COMMON /AREA1B/ XMINI,YMAXI,XMAXI,YMINI,IGEOM
      COMMON /AREA2/ NSET,MAXFRA,MAXINT,MAXELT,NUMINT,NUMNPT,NUMELT,
     *NOINT,IPRT
      COMMON /AREA4/ ATH(5),BTH(5),ATHP(5),BTHP(5),THDSTR(5),
     *NFRAC(5)
      COMMON /AREA7/ R(8),XB(8),YB(8),ISHAP(8),IBC(8)
      DIMENSION X1(MAXFRA,NSET),XORD(MAXINT),YORD(MAXINT)
      INTEGER INC1(MAXINT,2),INC2(MAXINT,2),IB(MAXINT)
      LOGICAL EFF(MAXINT),EFL
C
      II=1
      NOINT=0
C
C INITIALIZE EFF ARRAY.
      DO 10 LL=1,NUMINT
      IF (IB(LL) .EQ. 0) THEN
        EFF(LL)=.TRUE.
      ELSE
        IF(IBC(IB(LL)) .EQ. 0) THEN
          EFF(LL)=.FALSE.
        ELSE
          EFF(LL)=.TRUE.
        END IF
      END IF
   10 CONTINUE
C
C BEGINNING OF 3 NESTED DO-LOOPS (SETS > FRACTURES > INTERSECTIONS ).
   20 DO 100 I=1,NSET
        NFRA = NFRAC(I)
      DO 95 J=1,NFRA
           IF(X1(J,I) .GE. 999.0) GO TO 95
C
C COUNT THE NUMBER OF EFFECTIVE INTERSECTIONS (M) ON FRACTURE (J,I),
C UNTIL M=3.
C
      M = 1
      DO 40 LL=1,NUMINT
      IF(INC1(LL,2) .NE. J .AND. INC2(LL,2) .NE. J) GO TO 40
```

```
      IF(M .EQ. 3) GO TO 95
      IF(.NOT. EFF(LL))GO TO 40
      IF((INC1(LL,1) .EQ. I .AND. INC1(LL,2) .EQ. J)
     *.OR. (INC2(LL,1) .EQ. I .AND. INC2(LL,2) .EQ. J)) THEN
        INTER=LL
        M=M+1
      END IF
   40 CONTINUE
      IF(M .NE. 2) GO TO 95
C
C ASSIGN THE VALUE .FALSE. IN ARRAY EFF FOR AN INTERSECTION THAT IS
C HYDRAULICALLY INEFFECTIVE, I.E. WHEN THE CONSIDERED INTERSECTION
C IS THE ONLY INTERSECTION WITH EFF=.TRUE. FOR ANY ONE OF THE TWO
C INTERSECTING FRACTURES.
C
      LL = INTER
      EFF(LL)=.FALSE.
   95 CONTINUE
  100 CONTINUE
C
C SUM UP THE NOEFF ARRAY AND CHECK FOR INCREASE WITH RESPECT TO
C PREVIOUS ITERATION.
C
      NOINT2 = 0
      DO 110 LL=1,NUMINT
  110 IF(.NOT. EFF(LL)) NOINT2=NOINT2+1
      IF(NOINT .EQ. NOINT2)GO TO 140
  120 NOINT = NOINT2
      II = II+1
       GO TO 20
  140 WRITE(2,145) II,NOINT
  145 FORMAT(////,' AFTER ITERATION ',I3,'  OF COMPLETE LOOPING OVER ALL
     *THE FRACTURES,'/' THE TOTAL NUMBER OF NON-EFFECTIVE INTERSECTIONS
     *IS FOUND TO BE ',I4)
C
C  REARRANGE THE INTERSECTIONS ARRAYS, PLACING THE EFFECTIVE
C  INTERSECTIONS FIRST.
C
      M=1
      DO 250 LL=1,NUMINT
      IF(.NOT. EFF(LL))GO TO 250
      IF(M .EQ. LL) GO TO 240
      IBL = IB(LL)
      XL = XORD(LL)
      YL = YORD(LL)
      I11L = INC1(LL,1)
      I12L = INC1(LL,2)
      I21L = INC2(LL,1)
      I22L = INC2(LL,2)
      EFL = EFF(LL)
C
      IB(LL) = IB(M)
      XORD(LL) = XORD(M)
      YORD(LL) = YORD(M)
      INC1(LL,1) = INC1(M,1)
      INC1(LL,2) = INC1(M,2)
      INC2(LL,1) = INC2(M,1)
```

```
      INC2(LL,2) = INC2(M,2)
      EFF(LL) = EFF(M)
C
      IB(M) = IBL
      XORD(M) = XL
      YORD(M) = YL
      INC1(M,1) = I11L
      INC1(M,2) = I12L
      INC2(M,1) = I21L
      INC2(M,2) = I22L
      EFF(M) = EFL
C
  240 M = M+1
  250 CONTINUE
      NUMNPT = M-1
C
C PRINT OUT INTERSECTION DATA.
C
      CALL OUTPT1(XORD,YORD,IB,INC1,INC2)
C
      RETURN
      END
```

OUTPT1

```
C
C ***************************************************************************
C
C THE SUBROUTINE OUTPT1 CREATES BOTH A FORMATTED AND AN UNFORMATTED
C FILE CONTAINING THE INTERSECTION DATA.
C
      SUBROUTINE OUTPT1(XORD,YORD,IB,INC1,INC2)
C
      COMMON /AREA2/ NSET,MAXFRA,MAXINT,MAXELT,NUMINT,NUMNPT,NUMELT,
     *NOINT,IPRT
      COMMON /AREA3B/ TITLE(20)
      DIMENSION XORD(NUMNPT),YORD(NUMNPT)
      INTEGER INC1(MAXINT,2),INC2(MAXINT,2),IB(NUMNPT)
C (N.B. INC1 AND INC2 HAVE TWO COLUMNS : FIRST COLUMN CONTAINS
C       FRACTURE SET NUMBER, SECOND ONE THE FRACTURE NUMBER.
C       STORAGE IS BY COLUMN)
C
      WRITE(5) NUMNPT,MAXINT
      WRITE(5) XORD
      WRITE(5) YORD
      WRITE(5) IB
      WRITE(5) INC1
      WRITE(5) INC2
C
      WRITE(2,10) NUMINT,NUMNPT
   10 FORMAT(///,' TOTAL NUMBER OF INTERSECTIONS (NUMINT) :',I5,/,
     *' NUMBER OF EFFECTIVE INTERSECTIONS (NUMNPT) :',I5)
      IF(IPRT .EQ. 1)GO TO 100
C
      WRITE(15,18) TITLE
   18 FORMAT('1',//,1X,20A4,//,' FILE: NODFO.U15')
      WRITE(15,20) NUMNPT,NOINT,MAXINT
   20 FORMAT(////,25X,'INTERSECTION DATA',///,' NUMBER OF INTERSECTIONS',
     */10X,'EFFECTIVE :',T30,I5,' (NUMNPT)',/,10X,'NON-EFFECTIVE :',T30,
     *I5,' (NOINT)',/,10X,'MAXINT=',T30,I5,     //,' FILE CONTAINS :',/
     *10X,'XORD(NUMNPT), FORMAT(10F10.6)',/,10X,'YORD(NUMNPT), FORMAT(10
     *F10.6)',/,10X,'IB(NUMNPT), FORMAT(50I2)',/,10X,'INC1(MAXINT,2), '
     *,'FORMAT(20I5)',/,10X,'INC2(MAXINT,2), FORMAT(20I5)')
      WRITE(15,25) XORD
      WRITE(15,25) YORD
   25 FORMAT(10F10.6)
      WRITE(15,30) IB
   30 FORMAT(50I2)
      WRITE(15,35) INC1
      WRITE(15,35) INC2
   35 FORMAT(20I5)
C
  100 RETURN
      END
```

```
C  ***********************************************************************
C
C THE SUBROUTINE PLOTFR CREATES A PLOT OF THE LINE NETWORK.
C
      SUBROUTINE PLOTFR(PLTSIZ,TANTH,X1,Y1,X2,Y2,XORD,YORD)
C
      INTEGER THDSTR
      COMMON /AREA1A/ XMINO,YMAXO,XMAXO,YMINO
      COMMON /AREA1B/ XMINI,YMAXI,XMAXI,YMINI,IGEOM
      COMMON /AREA2/ NSET,MAXFRA,MAXINT,MAXELT,NUMINT,NUMNPT,NUMELT,
     *NOINT,IPRT
      COMMON /AREA3B/ TITLE(20)
      COMMON /AREA4/ ATH(5),BTH(5),ATHP(5),BTHP(5),THDSTR(5),
     *NFRAC(5)
      COMMON /AREA7/ R(8),XB(8),YB(8),ISHAP(8),IBC(8)
      DIMENSION TANTH(MAXFRA,NSET),X1(MAXFRA,NSET),
     *Y1(MAXFRA,NSET),X2(MAXFRA,NSET),Y2(MAXFRA,NSET)
      DIMENSION XORD(MAXINT),YORD(MAXINT)
      CHARACTER Z*8
C
C  INITIALIZE AND SCALE THE PLOT.
C
      WRITE(Z,5) TITLE(1)
    5 FORMAT(1A4)
      READ(Z,7) RNUM
    7 FORMAT(1X,F3.0)
C
      XLEN = (XMAXO-XMINO)
      YLEN = (YMAXO-YMINO)
      FCTR = (PLTSIZ-3.)/ XLEN
C
      DELTAV = 1.0
      XOR=2.0 - XMINO*FCTR
      YOR=2.0 - YMINO*FCTR
C
      CALL PLOTS(0,0,99)
      CALL PLOT(XOR,YOR,-3)
      CALL FACTOR(FCTR)
C
      XTITLE = (XMAXO+XMINO)*0.5 - 0.15*PLTSIZ/FCTR
      YTITLE = YMAXO + 0.05*PLTSIZ/FCTR
      SIZE = 0.025*PLTSIZ/FCTR
      CALL SYMBOL(XTITLE,YTITLE,SIZE,'REALIZATION ',0.0,12)
      CALL NUMBER(999.0,999.0,SIZE,RNUM,0.0,-1)
C
C DRAW THE EXTERNAL BOUNDARIES.
C
      CALL AXIS(XMINO,YMINO,'X-AXIS',-6,XLEN,0.0,XMINO,DELTAV)
      CALL AXIS(XMAXO,YMINO,6H      ,-1,YLEN,90.0,YMINO,DELTAV)
      CALL AXIS(XMINO,YMINO,'Y-AXIS',6,YLEN, 90.0,YMINO,DELTAV)
      CALL AXIS(XMINO,YMAXO,6H      ,0,XLEN,0.0,XMINO,DELTAV)
C
C  DRAW THE INTERNAL BOUNDARIES.
C
      IF(IGEOM .EQ. 1) THEN
```

```
        CALL PLOT(XMINI,YMINI,3)
        CALL PLOT(XMAXI,YMINI,2)
        CALL PLOT(XMAXI,YMAXI,2)
        CALL PLOT(XMINI,YMAXI,2)
        CALL PLOT(XMINI,YMINI,2)
      ELSE
        WRITE(2,25)
   25   FORMAT('1 ****** AVERTISSEMENT DE PLOTFR ******',///,
     *  'LA SOUS-ROUTINE CIRCL N''EXISTE PAS A L''U2AC :',
     *  'LES LIMITES CIRCULAIRES NE SONT PAS TRACEES',//)
        X=XB(1)
        Y=YB(1)
        CALL PLOT(X,Y,3)
        X=XB(3)
        Y=YB(3)
        CALL PLOT(X,Y,2)
C       RADIUS=R(3)
C       CALL CIRCL(X,Y,90.,0.,RADIUS,RADIUS,0.5)
           X=XB(4)
           Y=YB(4)
           CALL PLOT(X,Y,3)
        X=XB(6)
        Y=YB(6)
        CALL PLOT(X,Y,2)
C       RADIUS=R(6)
C       CALL CIRCL(X,Y,0.,90.,RADIUS,RADIUS,0.5)
      END IF
C
C  DRAW ALL THE LINE SEGMENTS (AND, OPTIONNALLY, WRITE DOWN THEIR NUMBER
C
      SIZE = 0.25 / FCTR
      MESS1 = 51
      MESS2 = 50
      DO 100  I=1,NSET
         NFRA = NFRAC(I)
      DO 90 J=1,NFRA
        IF(X1(J,I) .GE. 999.0) GO TO 90
      XX1 = X1(J,I)
      YY1 = Y1(J,I)
      XX2 = X2(J,I)
      YY2 = Y2(J,I)
      CALL PLOT(XX1,YY1,3)
      CALL PLOT(XX2,YY2,2)
      IF(NUMNPT .GT. 200 .OR. IPRT .EQ. 1)GO TO 90
C
        RJ = FLOAT(J)
        X = ( (XX1+XX2)*0.5)
        Y = ( (YY1+YY2)*0.5) - SIZE
        IF(ABS(TANTH(J,I)) .LT. 1) GO TO 30
        CALL SYMBOL(X,Y,SIZE,MESS1,0.0,-1)
        GO TO 40
   30   CALL SYMBOL(X,Y,SIZE,MESS2,0.0,-1)
   40   CALL NUMBER(999.0,999.0,SIZE,RJ,0.0,-1)
C
   90 CONTINUE
  100 CONTINUE
C
```

```
 PLOTFR

C  CIRCLE (AND, OPTIONALLY, WRITE THE NUMBER) OF THE EFFECTIVE
C  INTERSECTIONS, OR NODES.
C
      MESS = 1
      SIZE = 0.15 / FCTR
      DO 200 K=1,NUMNPT
      RNUM = K
      X =  XORD(K)
      Y =  YORD(K)
      CALL SYMBOL(X,Y,SIZE,MESS,0.0,-1)
      IF(NUMNPT .GT. 200 .OR. IPRT .EQ. 1)GO TO 200
C
        XN = X + SIZE
        YN = Y + SIZE
        CALL NUMBER(XN,YN,SIZE,RNUM,0.0,-1)
C
  200 CONTINUE
C
C
      ENDX = XMAX0+2.
      ENDY=-YOR/FCTR
      CALL PLOT(ENDX,0.00,999)
      RETURN
      END
```

```
C ***********************************************************************
C
C  THE SUBROUTINE ELTDEF DEFINES THE ELEMENTS IN ALL THE FRACTURES
C  WITH MORE THAN ONE EFFECTIVE INTERSECTIONS.
C
C                    LOCAL  VARIABLES
C
C APER(J) : APERTURE OF FRACTURE J IN THE SET BEING PROCESSED.
C INTER(M) : NUMBER OF THE INTERSECTION (GENERAL NUMBERING)
C             CORRESPONDING TO INTERSECTION M IN THE CURRENTLY
C             PROCESSED FRACTURE.
C MAXEFF : MAXIMUM NUMBER OF EFFECTIVE INTERSECTIONS ON A SINGLE
C          FRACTURE
C ML : COUNTER FOR FLOW-EFFECTIVE INTERSECTIONS ON LINE CURRENTLY
C      PROCESSED.
C NINTFR : NUMBER OF EFFECTIVE INTERSECTIONS ON THE CURRENTLY
C          PROCESSSED FRACTURE.
C
      SUBROUTINE ELTDEF(X1,Y1,X2,Y2,INC1,INC2,XORD,YORD,APER,CUT1,CUT2,
     *TH,THP,ELOR,ELORP,ELLEN,W,NOD1,NOD2,ISET)
C
      DOUBLE PRECISION SEED,PI,RAD
      INTEGER APDSTR,THDSTR
      COMMON /AREA1B/ XMINI,YMAXI,XMAXI,YMINI,IGEOM
      COMMON /AREA2/ NSET,MAXFRA,MAXINT,MAXELT,NUMINT,NUMNPT,NUMELT,
     *NOINT,IPRT
      COMMON /AREA3A/ SEED,PI,RAD,COEF
      COMMON /AREA4/ ATH(5),BTH(5),ATHP(5),BTHP(5),THDSTR(5),
     *NFRAC(5)
      COMMON /AREA6/ AAP(5),BAP(5),APDSTR(5),DAAP,NAAP,IAAP,NAP,IAP
C
      DIMENSION EFFLE(5),EFFVO(5),DEADLE(5),DEADVO(5)
      DIMENSION YINT(100),INTER(100)
      INTEGER CUT1(MAXFRA,NSET),CUT2(MAXFRA,NSET)
      DIMENSION TH(MAXFRA,NSET),THP(MAXFRA,NSET)
      DIMENSION X1(MAXFRA,NSET),Y1(MAXFRA,NSET),X2(MAXFRA,NSET),
     *Y2(MAXFRA,NSET)
      DIMENSION APER(MAXFRA)
      DIMENSION XORD(MAXINT),YORD(MAXINT)
      INTEGER INC1(MAXINT,2),INC2(MAXINT,2)
      DIMENSION ELLEN(MAXELT),W(MAXELT),ELOR(MAXELT),ELORP(MAXELT)
      INTEGER NOD1(MAXELT),NOD2(MAXELT),ISET(MAXELT)
      DATA MAXEFF/100/
C
C
      RADI = 1./RAD
      DO 350 IAAP=1,NAAP
      IF(IAAP .EQ. 1)GO TO 12
        DO 10 I=1,NSET
   10   AAP(I)=AAP(I)+DAAP
C
C PRINT OUT INPUT APERTURE DATA
   12 WRITE(2,14) IAAP,(I, I=1,NSET)
   14 FORMAT('1',///,10X,'APERTURE AND POROSITY DATA',
     *,/////////,10X,'INPUT DATA FOR APERTURE DISTRIBUTIONS',10X,
```

```
ELTDEF

     *'IAAP=',I4,///,' FRACTURE SET :',T25,5I10,///)
      WRITE(2,16) (APDSTR(I), I=1,NSET)
   16 FORMAT(//' DISTRIBUTION TYPE :',T25,5I10)
      WRITE(2,18) (AAP(I), I=1,NSET)
   18 FORMAT(/' DISTRIBUTION PARAMETERS',//,10X,'A :',T25,1P5E10.2)
      WRITE(2,20) (BAP(I), I=1,NSET)
   20 FORMAT(10X,'B :',T25,5F10.2)
C
      DO 300 IAP=1,NAP
        DO 22 I=1,5
        EFFLE(I)=0.0
        EFFVO(I)=0.0
        DEADLE(I)=0.0
        DEADVO(I)=0.0
   22   CONTINUE
        NUMELT=1
C
      DO 200 I=1,NSET
         NFRA = NFRAC(I)
C
C  GENERATE FRACTURE APERTURES.
C
      IF(APDSTR(I) .EQ. 2) GO TO 25
      DO 24 J=1,NFRA
   24 APER(J) = AAP(I)
      GO TO 30
   25 CALL GGNLG(SEED,NFRA,AAP(I),BAP(I),APER)
C
C
   30 DO 195 J=1,NFRA
      IF(X1(J,I) .GE. 999.0) GO TO 195
      APJ = APER(J)
C
C  MAKE A LIST OF ALL THE EFFECTIVE INTERSECTIONS ON FRACTURE (I,J).
C
      ML=1
      DO 60 M=1,NUMNPT
      IF(INC1(M,2) .NE. J .AND. INC2(M,2) .NE. J) GO TO 60
      IF((INC1(M,1) .NE. I .OR. INC1(M,2) .NE. J) .AND.
     *  (INC2(M,1) .NE. I .OR. INC2(M,2) .NE. J))GO TO 60
      INTER(ML) = M
      YINT(ML) = YCRD(M)
      ML = ML+1
      IF(ML .LE. MAXEFF) GO TO 60
      WRITE(2,55) ML,J,I
   55 FORMAT('1**** ERROR IN SUBROUTINE ELTDEF ****',//,
     *' THERE ARE AT LEAST ',I4,' EFFECTIVE INTERSECTIONS ON FRACTURE '
     *,I4,' OF SET ',I4,//,' INCREASE THE VALUE OF MAXEFF AND THE ',
     *'DIMENSION OF YINT,INTER AND NUM')
      STOP
   60 CONTINUE
      NINTER = ML-1
      IF(NINTER .GT. 1) GO TO 65
C
C  ADD THE LENGTH AND VOLUME OF FRACTURES WITH LESS THAN TWO
C  EFFECTIVES INTERSECTIONS TO THE VALUE OF DEADLE(I) AND DEADVO(I).
C
```

```
ELTDEF

      DXL=X1(J,I)-X2(J,I)
      DYL=Y1(J,I)-Y2(J,I)
      DEADL=SQRT(DXL*DXL+DYL*DYL)
      DEADLE(I) = DEADLE(I) + DEADL
      DEADVO(I) = DEADVO(I) + DEADL*APJ
      GO TO 195
C
C   ORDER THE INTERSECTIONS BY INCREASING Y VALUE.
C
   65 CALL VSRTR(YINT,NINTFR,INTER)
C
C   ADD THE LENGTH AND THE VOLUME OF DEAD END SEGMENT(S) OF FRACTURE
C   (I,J) TO THE VALUE OF DEADLE(I) AND DEADVO(I).
C
      IF(CUT1(J,I) .LT. 2) THEN
        I1=INTER(NINTFR)
        DX=X1(J,I)-XORD(I1)
        DY=Y1(J,I)-YORD(I1)
        DLE=SQRT(DX*DX+DY*DY)
        DEADLE(I)=DEADLE(I)+DLE
        DEADVO(I)=DEADVO(I)+DLE*APJ
      END IF
      IF(CUT2(J,I) .LT. 2) THEN
        I2=INTER(1)
        DX=X2(J,I)-XORD(I2)
        DY=Y2(J,I)-YORD(I2)
        DLE=SQRT(DX*DX+DY*DY)
        DEADLE(I)=DEADLE(I)+DLE
        DEADVO(I)=DEADVO(I)+DLE*APJ
      END IF
C
C   DEFINE AS AN  ELEMENT EVERY SEGMENT BETWEEN TWO CONSECUTIVE NODES.
C
  103 DO 150 M=2,NINTFR
      M1 = M-1
      DY=YINT(M)-YINT(M1)
      IF(DY .GT. 0.000001)GO TO 110
        WRITE(2,105) I,J,M1,YINT(M1),M,YINT(M)
  105   FORMAT('1',////,' **** JOB ABANDONED DUE TO ELEMENT PRACTICALLY'
     *,'PARALLEL TO X-AXIS',///,' SET',I3,';  FRACTURE',I5,//,
     *2(' INTERSECTION',I3,', Y=',T25,F10.7,/))
        STOP
C
  110 XL=SQRT((XORD(INTER(M))-XORD(INTER(M1)))**2 + DY*DY)
      EFFLE(I)=EFFLE(I)+XL
      EFFVO(I)=EFFVO(I)+XL*APJ
C
      NOD1(NUMELT)=INTER(M-1)
      NOD2(NUMELT)=INTER(M)
      ELLEN(NUMELT)=XL
      W(NUMELT)=APJ
      ISET(NUMELT)=I
      THL = ATAN(TH(J,I)) * RADI
       IF(THL .LE. 0.0) THL=180.+THL
       ELOR(NUMELT)=THL
      ELORP(NUMELT)=THP(J,I)
C
```

```
ELTDEF

      NUMELT=NUMELT+1
  150 CONTINUE
C
  195 CONTINUE
  200 CONTINUE
      NUMELT = NUMELT-1
      IF(NUMELT .LT. MAXELT)GO TO 204
      WRITE(2,202) NUMELT,MAXELT
  202 FORMAT('1*** ERROR CONCERNING MAXELT ***',///,' THERE ARE ',I4,
     *' ELEMENTS AND THERE IS MEMORY SPACE AVAILABLE FOR ',I4,
     *' (MAXELT)')
      STOP
C
C PRINT OUT ELEMENT DATA
  204 CALL OUTPT2(NOD1,NOD2,ISET,ELLEN,W,ELOR,ELORP)
C
C  PRINT OUT POROSITY DATA.
C
C
      WRITE(2,215) IAAP,IAP
  215 FORMAT(/////10X,'SUMMARY OF POROSITY DATA',5X,'IAAP=',I4,5X,'IAP='
     *,I4,///,5X,'SET',10X,'LENGTH'
     *,24X,'VOLUME',//,2(11X,'EFFECTIVE',6X,'DEAD'),/)
      WRITE(2,220) (I,EFFLE(I),DEADLE(I),EFFVO(I),DEADVO(I),I=1,NSET)
  220 FORMAT(5X,I3,0P2F11.2,10X,1P2E10.2)
C
  300 CONTINUE
  350 CONTINUE
      RETURN
      END
```

```
C
C ***************************************************************************
C
C THE SUBROUTINE OUTPT2 CREATES TWO PAIRS OF FILES : ONE PAIR CONTAINING
C THE ELEMENT INCIDENCES AND LENGTH, THE SECOND THE FRACTURE APERTURES.
C EACH PAIR CONSISTS OF ONE FORMATTED AND ONE UNFORMATTED FILE.
C
      SUBROUTINE OUTPT2(NOD1,NOD2,ISET,ELLEN,W,ELOR,ELORP)
C
      COMMON /AREA2/ NSET,MAXFRA,MAXINT,MAXELT,NUMINT,NUMNPT,NUMELT,
     *NOINT,IPRT
      COMMON /AREA39/ TITLE(20)
      COMMON /AREA6/ AAP(5),BAP(5),APDSTR(5),DAAP,NAAP,IAAP,NAP,IAP
      DIMENSION ELLEN(NUMELT),W(NUMELT),ELOR(NUMELT),ELORP(NUMELT)
      INTEGER NOD1(NUMELT),NOD2(NUMELT),ISET(NUMELT)
C
      IF(IAAP .GT. 1 .OR. IAP .GT. 1)GO TO 105
      WRITE(2,15) NUMELT
   15 FORMAT(//,' NUMBER OF ELEMENTS (NUMELT) :',I5)
      WRITE(6) NUMELT
      WRITE(6) NOD1
      WRITE(6) NOD2
      WRITE(6) ELLEN
      WRITE(6) ISET
      WRITE(6) ELOR
      WRITE(6) ELORP
      WRITE(7) NUMELT,NAAP,NAP
      IF(IPRT .EQ. 1)GO TO 105
C
      WRITE(16,50) TITLE,NUMELT
   50 FORMAT('1',//,1X,20A4,//,' FILE: ELEF01.U16',
     +///,25X,'ELEMENT DATA',///,'NUMBER OF ELEMENTS :'
     *,T30,I5,' (NUMELT)',///,' FILE CONTAINS :',//,10X,'NOD1(NUMELT), ',
     *'FORMAT(20I5)',//,10X,'NOD2(NUMELT), FORMAT(20I5)',//,
     *10X,'ISET(NUMELT), FORMAT(50I2)',//,
     * 10X,'ELLEN(NUMELT), FORMAT(1P10E10.3)',//,
     *10X,'ELOR(NUMELT), FORMAT(1P10E10.3)',//,
     *10X,'ELORP(NUMELT), FORMAT(1P10E10.3)')
      WRITE(16,60) NOD1
      WRITE(16,60) NOD2
   60 FORMAT(20I5)
      WRITE(16,62) ISET
   62 FORMAT(50I2)
      WRITE(16,70) ELLEN
   70 FORMAT(1P10E10.3)
      WRITE(16,75) ELOR
      WRITE(16,75) ELORP
   75 FORMAT(10F10.3)
C
      WRITE(17,100) TITLE,NUMELT,NAAP,NAP
  100 FORMAT('1',//,1X,20A4,//,' FILE: APEF01.U17',
     +///,25X,'FRACTURE APERTURES',///,' NUMBER OF ',
     *'ELEMENTS :',T30,I5,' (NUMELT)',//,' NUMBER OF DIFF. VAL. FOR ',
     *'FIRST PARAM (AAP) :',T50,I5,' (NAAP)',//,' NUMBER OF ',
     *'REALIZATIONS FOR EACH VAL. OF AAP :',T50,I5,' (NAP)',///,
```

```
OUTPT2

     *' FILE CONTAINS : NAAP*NAP ARRAYS W(NUMELT), FORMAT(1P10E10.3)')
  105 WRITE(7) W
      IF(IPRT .EQ. 1)GO TO 150
      WRITE(17,130) W
  130 FORMAT(1P10E10.3)
C
  150 RETURN
      END
```

G.2 Liste du programme APEGEN

APEGEN

```
      PROGRAM APEGEN
C*************************************************************************
C
C  PROGRAM APEGEN GENERATES FRACTURE APERTURE VALUES FOR THE FRACTURE
C  NETWORK GENERATED BY THE PROGRAM NETWRK
C
C  WRITTEN BY A. ROULEAU, DECEMBER 1984; UPDATE BY A. ROULEAU
C  AND R. MACLEOD, JUNE 1985.
C
C*************************************************************************
      INTEGER APDSTR
      DOUBLE PRECISION SEED
      COMMON /AREA1/ TITLE(20)
      COMMON /AREA2/ NUMNPT,NUMELT,IPRT
      COMMON /AREA3/ AAP(5),BAP(5),SEED,APDSTR,NSET
      COMMON /AREA4/ DAAP,NAAP,NAP,IAAP,IAP
C
C DIMENSION MAXELT=200
      INTEGER NOD1(200),NOD2(200),ISET(200)
      DIMENSION W(200),WW(200),ELLEN(200),ELOR(200),ELORP(200)
      DIMENSION P(200),Q(200),R(200)
C
C DIMENSION MAXNOD=150
      DIMENSION XORD(150),YORD(150)
C
C CONSTANTS
      MAXELT=200
      MAXNOD=150
C
      READ(21,3) TITLE
3     FORMAT(20A4)
C
C READ GENERAL INPUT DATA
      READ(21,5) IPRT,APDSTR,NAP,NAAP,DAAP,SEED,NSET,IOR
5     FORMAT(4I5,F10.0,F5.0,2I5)
      READ(21,8) (AAP(I),BAP(I),I=1,NSET)
8     FORMAT(2F10.0)
C
C READ NETWORK DATA
      READ(5) NUMNPT
      READ(6) NUMELT
      CALL INPT3(XORD,YORD,NOD1,NOD2,ISET,ELLEN,ELOR,ELORP,W)
      IF(IOR .EQ. 0) GO TO 600
C
C WITHIN THE FOLLOWING DO-LOOP (DO 500) , FOR EACH ELEMENT, THE
C COORDINATES OF THE CENTERPOINT IS DETERMINED AND THE ORIENTATION
C IS COMPUTED IN TERMS OF THE DIRECTION COSINES P,Q,R.
C THESE RESULTS ARE NOT USED IN THE PRESENT VERSION, BUT THEY
C WOULD BE USEFUL FOR INSTANCE TO DETERMINE APERTURE AS A FUNCTION
C OF FRACTURE ORIENTATION AND LOCATION.
C
C LOOP OVER ALL THE ELEMENTS
      DO 500 N=1,NUMELT
C
C COMPUTE DIRECTION COSINES
```

```
 APEGEN


       THN=ELOR(N)
       THPN=ELORP(N)
       ASIN=SIN(THPN)
       P(N)=COS(THPN)
       Q(N)=ASIN*SIN(THN)
       R(N)=-ASIN*COS(THN)
C CHECK ON DIRECTION COSINES
       PN=P(N)
       QN=Q(N)
       RN=R(N)
       P2=PN*PN
       Q2=QN*QN
       R2=RN*RN
       TOT=P2+Q2+R2
       WRITE(6,*) N,TOT
C
C DETERMINE THE COORDINATES OF THE ELEMENTS
C  FIRST END POINTS
       ND1=NOD1(N)
       ND2=NOD2(N)
       X1E=XORD(ND1)
       Y1E=YORD(ND1)
       X2E=XORD(ND2)
       Y2E=YORD(ND2)
C  THEN CENTERPOINT
       XCE = (X1E+X2E)/2.0
       YCE = (Y1E+Y2E)/2.0
C NOW IN CYLINDRICAL COORDINATES
       TANR=YCE/XCE
       THR=ATAN(TANR)
       RF=YCE/SIN(THR)
C
C MODIFY EXISTING APERTURE VALUE W(N) OR CREATE A NEW ONE
C
C  (NO OPERATION IS DONE HERE IN THE PRESENT VERSION OF APEGEN)
C
500    CONTINUE
       STOP
C
C CREATE AN ENTIRE ARRAY OF NEW APERTURES.
C
600    CALL APARR(W,WW,ISET)
C
       STOP
       END
```

INPT3

```
C*************************************************************************
C
C  SUBROUTINE INPT3 IS CALLED BY APEGEN-MAIN TO READ IN THE NODE
C  COORDINATE DATA (UNIT 9), THE ELEMENT INCIDENCE DATA (UNIT 10)
C  AND THE INITIAL FRACTURE APERTURE DATA (UNIT 11).
C
C*************************************************************************
C
      SUBROUTINE INPT3(XORD,YORD,NOD1,NOD2,ISET,ELLEN,ELOR,ELORP,W)
      COMMON /AREA2/ NUMNPT,NUMELT,IPRT
      DIMENSION XORD(NUMNPT),YORD(NUMNPT)
      DIMENSION ELLEN(NUMELT),W(NUMELT)
      INTEGER NOD1(NUMELT),NOD2(NUMELT),ISET(NUMELT)
C
      READ(5) XORD
      READ(5) YORD
C
      READ(6) NOD1
      READ(6) NOD2
      READ(6) ELLEN
      READ(6) ISET
      READ(6) ELOR
      READ(6) ELORP
C
      READ(7) DUMMY
      READ(7) W
C
      RETURN
      END
```

```
C*******************************************************************************
C
C  SUBROUTINE APARR CREATES AN ENTIRE ARRAY OF NEW APERTURE VALUES.
C
C*******************************************************************************
C
      SUBROUTINE APARR(W,WW,ISET)
      COMMON /AREA1/ TITLE(20)
      COMMON /AREA2/ NUMNPT,NUMELT,IPRT
      COMMON /AREA3/ AAP(5),BAP(5),SEED,APDSTR,NSET
      COMMON /AREA4/ DAAP,NAAP,NAP,IAAP,IAP
      INTEGER APDSTR
      INTEGER ISET(NUMELT)
      DOUBLE PRECISION SEED
      DIMENSION W(NUMELT),WW(NUMELT)
C
      WRITE(23,5) TITLE,NUMELT,APDSTR
5     FORMAT(1X,20A4,//,' FILE : APEF02.U23',/////,
     +25X,'FRACTURE APERTURES',///,5X,'NUMBER OF ELEMENTS : ',T30,I5,
     +' (NUMELT)',/,5X,'DISTRIBUTION TYPE : ',T30,I5)
      DO 350 IAAP=1,NAAP
      IF(IAAP .EQ. 1) GO TO 12
      DO 10 I=1,NSET
10    AAP(I)=AAP(I)+DAAP
C
12    DO 300 IAP=1,NAP
C
C PRINT OUT INPUT APERTURE DATA
      WRITE(23,14) IAAP,IAP,(I,I=1,NSET)
14    FORMAT(///,10X,'INPUT DATA FOR APERTURE DISTRIBUTIONS',10X,
     +'IAAP=',I4,'  IAP=',I4,///,' FRACTURE SET :',T25,5I10,///)
      WRITE(23,18) (AAP(I),I=1,NSET)
18    FORMAT(/,' DISTRIBUTION PARAMETERS',//,10X,'A :',T25,1P5E10.2)
      WRITE(23,20) (BAP(I), I=1,NSET)
20    FORMAT(10X,'B :',T25,5F10.2)
C
C GENERATE APERTURE VALUES
C
      IF (APDSTR .EQ. 2) GO TO 25
      DO 24 N=1,NUMELT
24    WW(N) = AAP(ISET(N))
      GO TO 30
25    DO 28 I=1,NSET
      CALL GGNLG(SEED,NUMELT,AAP(I),BAP(I),W)
      DO 28 N=1,NUMELT
28    IF(ISET(N) .EQ. I) WW(N)=W(N)
C
30    CALL OUTPT3(WW,ISET)
C
300   CONTINUE
C
350   CONTINUE
C
      RETURN
      END
```

OUTPT3

```
C*************************************************************************
C
C  SUBROUTINE OUTPT3 WRITES OUT A NEW FILE OF APERTURE VALUES
C  IN UNIT 22 (AND 23).
C
C*************************************************************************
C
      SUBROUTINE OUTPT3(WW,ISET)
      COMMON /AREA1/ TITLE(20)
      COMMON /AREA2/ NUMNPT,NUMELT,IPRT
      COMMON /AREA4/ DAAP,NAAP,NAP,IAAP,IAP
      DIMENSION WW(NUMELT)
      INTEGER ISET(NUMELT)
C
      IF (IAAP .GT. 1 .OR. IAP .GT. 1) GO TO 105
C
      WRITE(22) NUMELT,NAAP,NAP
105   WRITE(22) WW
C
      IF (IPRT .EQ. 1) GO TO 150
C
      WRITE(23,122)
122   FORMAT(///,20X,'ARRAY ISET, FORMAT(50I2)',///)
      WRITE(23,124) ISET
124   FORMAT(50I2)
      WRITE(23,128)
128   FORMAT(///,20X,'ARRAY WW, FORMAT(1P10E10.3)',///)
      WRITE(23,130) WW
130   FORMAT(1P10E10.3)
C
150   RETURN
      END
```

G.3 Liste du programme NETFLO

NETFLO

```
      PROGRAM NETFLO
C*************************************************************************
C
C THE PROGRAM NETFLO IS A NUMERICAL CODE THAT SOLVES THE STEADY-
C STATE EQUATIONS FOR FLUID FLOW IN THE NETWORK OF FRACTURES GENERATED
C BY THE PROGRAM NETWRK FORTRAN.
C
C WRITTEN BY A. ROULEAU, 1983; UPDATE BY A. ROULEAU AND D. LENTZ,
C APRIL 1985.
C
C*************************************************************************
C
      COMMON /AREA2/ MAXNOD,MAXELT,NUMNPT,NUMELT,NC,NF,NA,MAXA,
     *MAXCON,MAXFRE,IPRT,JAP
      COMMON /AREA4/ XB(8),YB(8),HDB(8),HDE(8),ISHAP(8),IBC(8),IGEOM,NBO
      COMMON /AREA5/ MAXEL2,MAXNO4,IDIFF
      COMMON /AREA6/ TITLE(20)
      COMMON /AREA8/ VISC,SPGR
C
C DIMENSION = MAXELT = 200
      INTEGER NOD1(200),NOD2(200),IDUMMY(200)
      DIMENSION W(200),E(200),ELLEN(200),ELOR(200)
C
C DIMENSION = MAXEL2 = 2 * MAXELT = 400
      INTEGER JT(400)
C
C DIMENSION = MAXNOD = 150
      INTEGER IB(150),LC(150),JMEM(150),NODNUM(150),NEWNOD(150),
     *JOINT(150),NEWJT(150)
      DIMENSION XORD(150),YORD(150),PHI(150)
      LOGICAL REOR(150)
C
C DIMENSION = MAXNO4 = 4 * MAXNOD = 600
      INTEGER MEMJT(600)
C
C DIMENSION = MAXFRE = 110
      INTEGER KDIAG(110)
C
C DIMENSION = MAXCON = MAXNOD - MAXFRE = 150-110=40
      DIMENSION PHIC(40)
C
C DIMENSION = MAXA = 2000
      DIMENSION A(2000)
C
C MAXELT + 1 = 200 + 1 = 201
      EQUIVALENCE (JT(1),NOD1),(JT(201),NOD2)
      EQUIVALENCE (ELOR,IDUMMY)
C
C CONSTANTS
C
      MAXELT = 200
      MAXNOD = 150
      MAXA = 2000
      MAXFRE = 110
      MAXCON = MAXNOD-MAXFRE
```

```
NETFLO

      MAXEL2=2*MAXELT
      MAXNO4=4*MAXNOD
      VISC=1.002/1000.
      SPGR=9800.
C
      READ(1,3) TITLE
    3 FORMAT(20A4)
C READ IN THE BOUNDARY CONDITIONS.
C
      READ(1,5)IGEOM,NAP,NAAP,IPRT
    5 FORMAT(25X,3I5,10X,I5,/)
        IF(IGEOM .EQ. 1)NBO=8
        IF(IGEOM .EQ. 2)NBO=6
      READ(1,7) (ISHAP(I),XB(I),YB(I),IBC(I),HDB(I),HDE(I),I=1,NBO)
    7 FORMAT(I5,5X,2F5.0,I5,2F5.0)
C
C READ IN THE NETWORK DATA
C
      READ(5) NUMNPT
      READ(6) DUMMY
      READ(7) NUMELT,NAAP,NAP
      CALL INPT1(XORD,YORD,IB,NOD1,NOD2,ELLEN,W,ELOR,IDUMMY)
C
C DEFINE THE BOUNDARY CONDITION FOR ALL THE NODES LOCATED ON A BOUNDARY
C
      CALL BOUND(IB,PHI,XORD,YORD)
C
C OPTIMIZE THE NODE NUMBERING AND DEFINE THE CONDENSATION CODE FOR
C SPARSE MATRIX STORAGE.
C
      CALL OPSTOR(IB,LC,KDIAG,JT,JMEM,MEMJT,NODNUM,NEWNOD,JOINT,NEWJT)
C
C SET UP THE MATRICES USING THE NODAL FLOW EQUATIONS, AND SOLVE FOR THE
C UNKNOWN HEAD VALUES (PHI) USING THE CHOLESKI ALGORITHM.
C
      NAPER=NAP+NAAP
      DO 200 JAP=1,NAPER
      IF(JAP .EQ. 1)GO TO 30
      CALL INPT2(W)
   30 CALL SOLPHI(IB,LC,KDIAG,NOD1,NOD2,XORD,YORD,PHI,PHIC,W,A,
     *E,NODNUM,NEWNOD,REOR,ELLEN)
C
C CALCULATE FLOW IN EACH ELEMENT AND TOTAL FLOW THROUGH THE MODEL.
C
      CALL FLOCAL(NOD1,NOD2,W,ELLEN,E,PHI,IB,ELOR)
C
  200 CONTINUE
      STOP
      END
```

```
C
C ***************************************************************************
C
C THE SUBROUTINE INPT1 READS IN THE LINE NETWRK AND FRACTURE APERTURE
C DATA FROM UNFORMATTED FILES CREATED BY NETWRK.
C
      SUBROUTINE INPT1(XORD,YORD,IB,NOD1,NOD2,ELLEN,w,ELOR,IDUMMY)
C
      COMMON /AREA2/ MAXNOD,MAXELT,NUMNPT,NUMELT,NC,NF,NA,MAXA,
     *MAXCON,MAXFRE,IPRT,JAP
      DIMENSION XORD(NUMNPT),YORD(NUMNPT)
      INTEGER IB(NUMNPT)
      DIMENSION ELLEN(NUMELT),w(NUMELT),ELOR(NUMELT)
      INTEGER NOD1(NUMELT),NOD2(NUMELT),IDUMMY(NUMELT)
C
      READ(5) XORD
      READ(5) YORD
      READ(5) IB
C
      READ(6) NOD1
      READ(6) NOD2
      READ(6) ELLEN
      READ(6) IDUMMY
      READ(6) ELOR
C
      ENTRY INPT2(w)
      READ(7) w
C
      RETURN
      END
```

```
C*********************************************************************
C
C THE SUBROUTINE BOUND DEFINES THE BOUNDARY CONDITIONS FOR ALL THE NODES
C LOCATED ON THE BOUNDARIES OF THE FLOW MODEL.
C
C                       LOCAL VARIABLES
C
C BEG(L) : ORDINATE (ABSCISSA) OF BEGINNING POINT OF BOUNDARY L, IF
C THIS BOUNDARY IS PARALLEL TO Y-AXIS (TO X-AXIS).
C END(L) : SAME AS BEG(L) FOR END POINT.
C GRAD(L) : IF IBC(L)=2, HEAD GRADIENT ALONG BOUNDARY L
C           IF IBC(L)=3, HEAD DIFF. / DIFF. IN LOG OF THE EXTREM. POINT
C
      SUBROUTINE BOUND(IB,PHI,XORD,YORD)
      COMMON /AREA2/ MAXNOD,MAXELT,NUMNPT,NUMELT,NC,NF,NA,MAXA,
     *MAXCON,MAXFRE,IPRT,JAP
      COMMON /AREA4/ XB(8),YB(3),HDB(8),HDE(8),ISHAP(8),IBC(8),IGEOM,NBO
      COMMON /AREA6/ TITLE(20)
      INTEGER IB(MAXNOD)
      DIMENSION XORD(MAXNOD),YORD(MAXNOD),PHI(MAXNOD)
      DIMENSION BEG(8),END(8),GRAD(8)
C
C DETERMINE END POINT (END( ) ) FOR EACH BOUNDARY
C
      IF(ISHAP(2) .EQ. 0) THEN
        END(1)=YB(3)
      ELSE
        END(1)=YB(2)
        END(2)=YB(3)
      END IF
      IF(IGEOM .EQ. 1)GO TO 10
C
C IF CIRCULAR BOUNDARIES
      IF(ISHAP(5) .EQ. 0) THEN
        END(4)=XB(6)
      ELSE
        END(4)=XB(5)
        END(5)=XB(6)
      END IF
      GO TO 20
C
   10 IF(ISHAP(4) .EQ. 0) THEN
        END(3)=XB(5)
      ELSE
        END(3)=XB(4)
        END(4)=XB(5)
      END IF
      IF(ISHAP(6) .EQ. 0) THEN
        END(5)=YB(7)
      ELSE
        END(5)=YB(6)
        END(6)=YB(7)
      END IF
      IF(ISHAP(8) .EQ. 0) THEN
        END(7)=XB(1)
```

```
        ELSE
          END(7)=XB(8)
          END(8)=XB(1)
        END IF
C
C CALCULATE VALUE OF CONSTANTS BEG( ) AND GRAD( ) FOR EACH BOUNDARY
C FOR WHICH IBC=2 OR 3.
C
   20 DO 50 L=1,NBC
        M1=IBC(L)
        IF(M1 .LT. 2) GO TO 50
        DHD=HDE(L)-HDB(L)
        IF(IGEOM .EQ. 2)GO TO 25
C RECTANGULAR BOUNDARIES
        IF(L .EQ. 1 .OR. L .EQ. 2 .OR. L .EQ. 5 .OR. L .EQ. 6) THEN
          BEG(L)=YB(L)
        ELSE
          BEG(L)=XB(L)
        END IF
        GO TO 30
C CIRCULAR BOUNDARIES
   25 IF(L .EQ. 1 .OR. L .EQ. 2) THEN
          BEG(L)=YB(L)
        ELSE
          BEG(L)=XB(L)
        END IF
C
   30 DLE=END(L)-BEG(L)
        IF(M1 .EQ. 2) THEN
          GRAD(L)=DHD/DLE
        ELSE
          GRAD(L)=DHD/ALOG(END(L)/BEG(L))
        END IF
   50 CONTINUE
C
C DETERMINE HEAD VALUE FOR EACH NODE LOCATED ON A BOUNDARY FOR WHICH
C IBC( )=1, 2 OR 3.
C
        NC=0
        DO 100 M=1,NUMNPT
        IBM=IB(M)
        IF(IBM .EQ. 0)GO TO 100
        M1=IBC(IBM)
        IF(M1 .EQ. 0)GO TO 100
        IF(M1 .EQ. 1) THEN
          PHI(M)=HDB(IBM)
        ELSE
C ASSIGN VALUE FOR XORY
          IF(IGEOM .EQ. 1) THEN
            IF(IBM .EQ. 1 .OR. IBM .EQ.2.OR.IBM.EQ.5.OR.IBM.EQ.6) THEN
              XORY=YCRD(M)
            ELSE
              XORY=XCRD(M)
            END IF
          ELSE
            IF(IBM .EQ. 1 .OR. IBM .EQ. 2) THEN
              XORY=YCRD(M)
```

```
 BOUND

          ELSE
            XORY=XCRD(M)
          END IF
        END IF
C
        IF(M1 .EQ. 2) THEN
          PHI(M)=HDB(IBM) + GRAD(IBM)*(XORY-BEG(IBM))
        ELSE
          PHI(M)=HDB(IBM) + GRAD(IBM)*ALOG(XORY/BEG(IBM))
        END IF
      END IF
      NC=NC+1
  100 CONTINUE
      NF=NUMNPT-NC
C
C PRINT OUT BOUNDARY DATA
C
      WRITE(31,200) TITLE
200   FORMAT('1',//,1X,20A4,//,' FILE: SUMFLO.U31',
     +////,10X,'OUTPUT FROM SUBROUTINE BOUND',///)
      WRITE(31,340)NUMNPT,NF,NC
  340 FORMAT(//,10X,'TOTAL NUMBER OF NODES',T40,I5,' (NUMNPT)',//,
     *10X,'NUMBER OF FREE NODES',T40,I5,' (NF)',//,
     *10X,'NUMBER OF CONSTRAINED NODES',T40,I5,' (NC)',////)
C
C PRINT OUT CONSTANTS FOR EACH BOUNDARY
      WRITE(31,350)
  350 FORMAT('  BDRY NO   IBC',7X,'BEG',10X,'END',8X,'''GRADIENT''',//)
      WRITE(31,355) (L,IBC(L),BEG(L),END(L),GRAD(L), L=1,NBD)
  355 FORMAT(2I7,3F13.3)
      IF(NF .LE. MAXFRE .AND. NC .LE. MAXCON)GO TO 370
C
      WRITE(31,365)NF,MAXFRE,NC,MAXCON
  365 FORMAT('1**** ERROR CONCERNING MAXFRE OR MAXCON ****',///,' THERE A
     *RE ',I5,' FREE NODES, AND MAXFRE IS',I5,///,' THERE ARE ',I4,' CONS
     *TRAINED NODES, AND MAXCON IS ',I4)
      STOP
  370 RETURN
      END
```

```
C***********************************************************************
C
C THE SUBROUTINE OPSTOR OPTIMIZES THE NODE NUMBERING AND DEFINES THE
C CONDENSATION CODES FOR MATRIX STORAGE.
C OPSTOR CALLS THE SUBROUTINES SETUP AND OPTNUM.
C
      SUBROUTINE OPSTOR(IB,LC,KDIAG,JT,JMEM,MEMJT,NODNUM,NEWNOD,JOINT,
     *NEWJT)
      COMMON /AREA2/ MAXNOD,MAXELT,NUMNPT,NUMELT,NC,NF,NA,MAXA,
     *MAXCON,MAXFRE,IPRT,JAP
      COMMON /AREA4/ XB(8),YB(8),HDB(8),HDE(8),ISHAP(8),IBC(8),IGEOM,NBO
      COMMON /AREA5/ MAXEL2,MAXNO4,IDIFF
      COMMON /AREA6/ TITLE(20)
      INTEGER KDIAG(MAXFRE)
      INTEGER IB(MAXNOD),LC(MAXNOD),JT(MAXEL2),
     *JMEM(MAXNOD),MEMJT(MAXNO4),NODNUM(MAXNOD),NEWNOD(MAXNOD),
     *JOINT(MAXNOD),NEWJT(MAXNOD)
C
C DEFINE THE TOPOLOGY OF THE ORIGINAL NODE NUMBERING.
C
      CALL SETUP(JT,JMEM,MEMJT)
C
C PRINT OUT THE DATA CONCERNING THE INITIAL NODE NUMBERING.
      WRITE(31,20) IDIFF
   20 FORMAT(//,10X,'OUTPUT FROM SETUP',//,' INITIAL IDIFF=',I5,/)
      IF(IPRT .LT. 3)GO TO 38
C
      WRITE(34,22) TITLE,IDIFF
22    FORMAT('1',/,1X,20A4,/,' FILE: FLOMAT.U34',
     +/,9X,'OUTPUT FROM SETUP',/,' INITIAL IDIFF=',I5,/)
      WRITE(34,24)
   24 FORMAT(10X,'FORMAT(6(I      JT(I)))',//)
      WRITE(34,26)(I,JT(I), I=1,MAXEL2)
   26 FORMAT(6(5X,2I5))
      WRITE(34,28) (I,JMEM(I),I=1,NUMNPT)
   28 FORMAT(////,10X,'FORMAT(6(I      JMEM(I)))',//,6(2I5,5X))
      WRITE(34,33)
   33 FORMAT(///,10X,'FORMAT(I    4 NODES RELATED TO NODE I, I.E. MEMJT(
     *4*(I-1) +1 TO +4))',/)
      DO 37 I=1,NUMNPT
        I4=4*(I-1)
        WRITE(34,36)I,MEMJT(I4+1),MEMJT(I4+2),MEMJT(I4+3),MEMJT(I4+4)
   36   FORMAT(  I5,2X,4I5,5X)
   37 CONTINUE
C
C RENUMBER THE NODES IN ORDER TO MINIMIZE THE BANDWIDTH OF THE MATRIX A
C
   38 CALL OPTNUM(JT,JMEM,MEMJT,NODNUM,NEWNOD,JOINT,NEWJT)
C
C DEFINE THE CONDENSATION CODE (LC) FOR MATRIX PARTITIONING, AND
C DETERMINE THE VALUES IN THE KDIAG ARRAY FOR STORAGE OF SUBMATRIX A.
C
      LC(1)=0
      M6=IB(NEWNOD(1))
      KDIAG(1)=1
```

```
      IF(M6 .EQ. 0)GO TO 50
      IF(IBC(M6) .GT. 0)LC(1)=1
   50 DO 100 M=2,NUMNPT
      M5=NEWNOD(M)
      M7=IB(M5)
C
C CALCULATE LC(M)
      K=0
      IF(M7 .EQ. 0)GO TO 53
      IF(IBC(M7).GT. 0)K=1
   53 LC(M)=LC(M-1)+K
      LCM=LC(M)
      IF(M7 .EQ. 0)GO TO 54
      IF(IBC(M7).GT. 0)GO TO 100
C
C CALCULATE KDIAG FOR NODE M, IF M IS A FREE NODE.
C FIRST, FIND WHICH ONE OF THE NODES RELATED TO NODE M HAS THE MINIMUM
C INDEX.
   54 NREL=JMEM(M5)
      MINREL=NODNUM(MEMJT(M5*4 - 3) )
      IF(NREL .EQ. 1)GO TO 60
      DO 55 M1=2,NREL
        M2=M5*4-4+M1
        M8=NODNUM(MEMJT(M2))
        IF(MINREL .GT. M8)MINREL=M8
   55   CONTINUE
   60 M3=M-LCM
C M3 INDICATES THE LOCATION OF NODE M ON THE DIAGONAL OF THE SUBMATRIX
C OF FREE NODES (A).
      IF(M3 .LT. 2)GO TO 100
      IF(MINREL .LT. M)GO TO 80
      KDIAG(M3)=KDIAG(M3-1)+1
      GO TO 100
   80 KDIAG(M3)=KDIAG(M3-1)+(M-MINREL)-(LCM-LC(MINREL))+1
  100 CONTINUE
      IF(LC(NUMNPT) .EQ. NC)GO TO 105
      WRITE(31,103) LC(NUMNPT),NC
  103 FORMAT('1**** ERROR CONCERNING CONSTRAINED NODES ****',///,
     *' LC(NUMNPT)=',I4,' NC=',I4)
      STOP
  105 CONTINUE
C
      NA=KDIAG(NF)
C
C PRINT OUT LC AND KDIAG ARRAYS
      WRITE(31,106)NA
106   FORMAT(///,10X,'RESULTS FROM OPSTOR',///,' SIZE OF MATRIX A (NA):',
     *I7)
      IF(IPRT .LT. 3)GO TO 120
C
      WRITE(34,108)
  108 FORMAT(///,10X,'CONDENSATION CODE (LC)', FORMAT(6(I    LC(I)))',//)
      WRITE(34,109) (I,LC(I),I=1,NUMNPT)
  109 FORMAT(6(2I5,5X))
      WRITE(34,112)
  112 FORMAT(///,10X,'VARIABLE BANDWIDTH STORAGE CODE (KDIAG)', FORMAT(6(I
     *  KDIAG(I)))',///)
```

```
OPSTOR


      WRITE(34,115) (I,KDIAG(I),I=1,NF)
  115 FORMAT(6(2I7,3X))
  120 IF(NA .LE. MAXA)GO TO 140
C
      WRITE(31,130)NA,MAXA
  130 FORMAT('1**** ERROR CONCERNING MAXA *****',///,' THE SIZE OF MATRIX
     * A (NA) SHOULD BE ',I8,' , BUT MAXA IS ',I8)
      STOP
  140 RETURN
      END
```

SETUP

```
C
C*****************************************************************************
C
C   THE SUBROUTINE SETUP GENERATES A LIST SHOWING THE RELATIONSHIP THE
C   NODES HAVE WITH EACH OTHER, AND COMPUTES THE ORIGINAL MATRIX BAND-
C   WIDTH (ADAPTED FROM COLLINS, 1973)
C
      SUBROUTINE SETUP(JT,JMEM,MEMJT)
      COMMON /AREA2/ MAXNOD,MAXELT,NODES,LMENTS,NC,NF,NA,MAXA,
     *MAXCON,MAXFRE,IPRT,JAP
      COMMON /AREA3/ MAXEL2,MAXNO4,IDIFF
      INTEGER JT(MAXEL2),JMEM(MAXNOD),MEMJT(MAXNO4)
C
      IDIFF=0
      DO 10 J=1,NODES
   10 JMEM(J)=0
      DO 60 J=1,LMENTS
      DO 50 I=1,2
      JNTI=JT(MAXELT*(I-1)+J)
      JSUB=(JNTI-1)*4
      DO 40 II=1,2
      IF(II.EQ.I)GOTO40
      JJT=JT(MAXELT*(II-1)+J)
      IF(JJT.EQ.0)GOTO50
      MEM1=JMEM(JNTI)
      IF(MEM1.EQ.0)GOTO30
      DO 20 III=1,MEM1
      IF(MEMJT(JSUB+III).EQ.JJT)GOTO40
   20 CONTINUE
   30 JMEM(JNTI)=JMEM(JNTI)+1
      MEMJT(JSUB+JMEM(JNTI))=JJT
      IF(IABS(JNTI-JJT).GT.IDIFF)IDIFF=IABS(JNTI-JJT)
   40 CONTINUE
   50 CONTINUE
   60 CONTINUE
      RETURN
      END
```

OPTNUM

```
C
C ***************************************************************
C
C  THE SUBROUTINE OPTNUM RENUMBERS THE NODES IN ORDER TO MINIMIZE
C  THE BAND-WIDTH OF THE MATRIX (MODIFIED FROM COLLINS, 1973).
C
      SUBROUTINE OPTNUM(JT,JMEM,MEMJT,NODNUM,NEWNOD,JOINT,NEWJT)
      COMMON /AREA2/ MAXNOD,MAXELT,NODES,LMENTS,NC,NF,NA,MAXA,
     *MAXCON,MAXFRE,IPRT,JAP
      COMMON /AREA5/ MAXEL2,MAXNO4,IDIFF
      INTEGER JOINT(MAXNOD),NEWJT(MAXNOD)
      INTEGER JT(MAXEL2),JMEM(MAXNOD),MEMJT(MAXNO4),NODNUM(MAXNOD),
     *NEWNOD(MAXNOD)
C
      IDIFF1=IDIFF
      MINMAX=IDIFF
      DO 60 IK=1,NODES
      DO 20 J=1,NODES
      JOINT(J)=0
   20 NEWJT(J)=0
      MAX=0
      I=1
      NEWJT(1)=IK
      JOINT(IK)=1
      K=1
   30 K4=JMEM(NEWJT(I))
      IF(K4.EQ.0)GOTO45
      JSUB=(NEWJT(I)-1)*4
      DO 40 JJ=1,K4
      K5=MEMJT(JSUB+JJ)
      IF(JOINT(K5).GT.0)GOTO40
      K=K+1
      NEWJT(K)=K5
      JOINT(K5)=K
      IDIFF=IABS(I-K)
      IF(IDIFF.GE.MINMAX)GOTO60
      IF(IDIFF.GT.MAX)MAX=IDIFF
   40 CONTINUE
      IF(K.EQ.NODES)GOTO50
      IF(K.GT.I)GOTO45
C
C IN CASE OF A DISCONNECTED NETWORK ....
C
         DO 42 J=1,NODES
         IF(JOINT(J).EQ.0)GOTO43
   42    CONTINUE
         GO TO 45
   43    K=K+1
         NEWJT(K)=J
         JOINT(J)=K
C
   45 I=I+1
      GOTO30
   50 MINMAX=MAX
```

```
 OPTNUM


C
      DO 55 J=1,NODES
      NEWNOD(J)=NEWJT(J)
   55 NODNUM(J)=JOINT(J)
   60 CONTINUE
C
      IF(IDIFF .LT. IDIFF1) GO TO 85
C
C IN CASE WHERE NO RENUMBERING IS REQUIRED
      DO 70 M=1,NODES
      NEWNOD(M)=M
70    NODNUM(M)=M
C PRINT OUT THE FINAL NODE NUMBERING
C
85    WRITE(31,90) IDIFF
   90 FORMAT(////,10X,'OUTPUT FROM OPTNUM',///,' FINAL IDIFF=',I5,///)
      IF(IPRT .LT. 3)GO TO 150
C
      WRITE(34,105)
  105 FORMAT(////,10X,'FINAL NODE RENUMBERING',///,
     *10X,'FORMAT(5(I   <==NEWNOD   ==>NODNUM))',///)
      WRITE(34,112)(I,NEWNOD(I),NODNUM(I) , I=1,NODES)
  112 FORMAT(5(3I5,5X))
C
  150 RETURN
      END
```

```
  SOLPHI


C*********************************************************************
C
C  THE SUBROUTINE SOLPHI CALCULATES THE VALUES IN THE SUBMATRICES
C  A AND PHIC, AND SOLVES THE SYSTEM FOR THE UNKNOWN VALUES OF PHI
C  USING THE CHOLESKY METHOD.
C  SOLPHI CALLS THE SUBROUTINES DECOMP, FORW AND BACK.
C
      SUBROUTINE SOLPHI(IB,LC,KDIAG,NOD1,NOD2,XORD,YORD,PHI,PHIC,W,A,
     *E,NODNUM,NEWNOD,REOR,ELLEN)
      COMMON /AREA2/ MAXNOD,MAXELT,NUMNPT,NUMELT,NC,NF,NA,MAXA,
     *MAXCON,MAXFRE,IPRT,JAP
      COMMON /AREA4/ XB(8),YB(8),HDB(8),HDE(8),ISHAP(8),IBC(8),IGEOM,NBO
      COMMON /AREA6/ TITLE(20)
      COMMON /AREA8/ VISC,SPGR
      INTEGER KDIAG(MAXFRE)
      INTEGER IB(MAXNOD),LC(MAXNOD),NODNUM(MAXNOD),
     *NEWNOD(MAXNOD),NOD1(MAXELT),NOD2(MAXELT)
      DIMENSION XORD(MAXNOD),YORD(MAXNOD),PHI(MAXNOD)
      DIMENSION E(MAXELT),ELLEN(MAXELT),W(MAXELT)
      DIMENSION IBCO(2),INEW(2)
      DIMENSION PHIC(MAXCON)
      DIMENSION A(MAXA)
      LOGICAL REOR(MAXNOD)
C
C  CONSTANTS
      FLUID=SPGR/(12*VISC)
C
C  CLEAR ARRAY A
      DO 5 I=1,MAXA
    5 A(I)=0.0
      IF(JAP .GT. 1)GO TO 13
C
C  SET UP CONSTRAINED HEAD VECTOR (PHIC)
      DO 10 I=1,NUMNPT
        M9=NEWNOD(I)
        M11=IB(M9)
        IF(M11 .EQ. 0)GO TO 10
        IF(IBC(M11) .EQ. 0)GO TO 10
        PHIC(LC(I))=PHI(M9)
   10 CONTINUE
C
C  CLEAR ARRAY PHI
   13 DO 15 I=1,NF
   15 PHI(I)=0.0
C
C  CALCULATE THE COEFFICIENT OF PHI FOR EACH ELEMENT AND SET UP THE
C  SUBMATRIX A (FOR TOTALLY FREE ELEMENTS).
C
      DO 250 N=1,NUMELT
C
C  CALCULATE COEFFICIENT E FOR ELEMENT N
      IOLD=NOD1(N)
      IBO=IB(IOLD)
      IBCO(1)=0
      IF(IBO .EQ. 0)GO TO 19
```

```
      IBCO(1)=IBC(IBO)
   18 INEW(1)=NODNUM(IOLD)
C
      IOLD=NOD2(N)
      IBO=IB(IOLD)
      IBCO(2)=0
      IF(IBO .EQ. 0)GO TO 20
      IBCO(2)=IBC(IBO)
   20 INEW(2)=NODNUM(IOLD)
C
      E(N)=(W(N)**3)*FLUID / ELLEN(N)
C
C ASSIGN THE VALUE 0, OR 1, OR 2, TO IB2, IF 0 NODE, 1 NODE, OR BOTH
C NODES ARE CONSTRAINED, RESPECTIVELY.
      IF(IBCO(1) .EQ. 0 .AND. IBCO(2) .EQ. 0)GO TO 30
      IF(IBCO(1) .EQ. 0 .OR. IBCO(2) .EQ. 0)GO TO 25
      IB2=2
      GO TO 35
   25 IB2=1
      GO TO 35
   30 IB2=0
   35 CONTINUE
C
C INSERT E INTO THE GLOBAL COEFFICIENT SUBMATRICES A OR AC.
C
      II=MAXO(INEW(1),INEW(2))
      JJ=MINO(INEW(1),INEW(2))
C
C SUBMATRIX A : LINEAR ARRAY AND VARIABLE BANDWIDTH STORAGE.
      DO 140 I=1,2
      IF(IBCO(I) .GT. 0)GO TO 140
      M4=KDIAG(INEW(I)-LC(INEW(I)))
      A(M4)=E(N)+A(M4)
  140 CONTINUE
      IF(IB2-1)160,170,250
  160 M5=KDIAG(II-LC(II)) - (II-JJ) + (LC(II)-LC(JJ))
      IF(M5 .LT. 1)GO TO 165
      A(M5)=-E(N)
      GO TO 250
  165 WRITE(31,166)M5
  166 FORMAT('1**** ERROR IN MATRIX A ********',///,' M5 = ',I6,' (???)')
      STOP
C
C AUGMENTED FLUX VECTOR (I.E. ENTRIES OF NODE CONDUCTANCE MATRIX FOR
C PARTIALLY CONSTRAINED ELEMENTS MULTIPLIED BY CORRESPONDING ENTRIES
C OF VECTOR PHIC )   STORED IN VECTOR PHI, WHICH HAS BEEN CLEARED
C PREVIOUSLY.
C
  170 DO 190 I=1,2
      IF(IBCO(I) .GT. 0)GO TO 190
      M6=INEW(I)-LC(INEW(I))
      II=IABS(I-3)
      M7=LC(INEW(II))
  190 CONTINUE
      AC=E(N)
      PHI(M6)=AC*PHIC(M7) + PHI(M6)
  250 CONTINUE
```

```
      IF(IPRT .LT. 3)GO TO 330
C
C PRINT OUT THE MATRICES TO BE SOLVED
C
      WRITE(34,295) JAP
  295 FORMAT(//////,'1 MATRICES A, ? (STORED IN ARRAY PHI), AND PHIC, BEF
     *ORE DECOMPOSITION',5X,'JAP=',I4,////,
     *5X,'MATRIX A',8X,'FORMAT(5(I    A(I)))',///)
      WRITE(34,301) (I,A(I),I=1,NA)
  301 FORMAT(5(4X,I3,1PE13.3))
C
      WRITE(34,313)JAP
  313 FORMAT(////9X,'VECTOR ''B'', STORED IN ARRAY PHI',10X,'FORMAT(5(I
     *  PHI(I)))',5X,'JAP=',I4,///)
      WRITE(34,316) (I,PHI(I),I=1,NF)
  316 FORMAT(5(I5,1PE12.3))
C
      WRITE(34,318) JAP
  318 FORMAT(////9X,'VECTOR PHIC',10X,'FORMAT(5(I    PHIC(I)))',5X,'JAP=',
     *I4,///)
      WRITE(34,321) (I,PHIC(I),I=1,NC)
  321 FORMAT(5(I5,1PE12.3))
C
C CHOLESKI DECOMPOSITION.
C
  330 CALL DECOMP(A,KDIAG)
C
C SOLVE FOR PHI, IN TWO STEPS:FORWARD-SUBSTITUTION AND BACKSUBSTITUTION
      CALL FORW(A,KDIAG,PHI)
C
      CALL BACK(A,KDIAG,PHI)
C
C EXPAND HEAD VECTOR
C
      DO 400 MM=1,NUMNPT
      M=NUMNPT-MM+1
      M10=I8(NEWNOD(M))
      IF(M10 .EQ. 0)GO TO 340
      IF(I8C(M10) .GT. 0)GO TO 350
  340 K=M-LC(M)
      PHI(M)=PHI(K)
      GO TO 400
  350 PHI(M)=PHIC(LC(M))
  400 CONTINUE
C
C REORDER PHI ARRAY IN INITIAL NODE ORDER.
C
      DO 410 M=1,NUMNPT
  410 REOR(M)=.FALSE.
      DO 450 M=1,NUMNPT
      IF(REOR(M))GO TO 450
      PHIM=PHI(M)
      MA=NEWNOD(M)
  420 PHIMA=PHI(MA)
      REOR(MA)=.TRUE.
      PHI(MA)=PHIM
      MA=NEWNOD(MA)
```

```
      IF(REOR(MA))GO TO 450
      PHIM=PHIMA
      GO TO 420
  450 CONTINUE
C
      IF(IPRT .LT. 2)GO TO 500
C
C PRINT THE RESULTS OF FLOW CALCULATION.
      WRITE(33,470) TITLE,JAP
  470 FORMAT('1',//,1X,20A4,//,' FILE: FLOALL.U33',
     +///,10X,'RESULTS OF FLOW CALCULATIONS,  JAP=',I4,///
     *10X,'HEAD AT NODAL POINTS',///,
     *10X,'FORMAT(3(I    IB(I)    PHI(I)))',///)
      WRITE(33,480)(M,IB(M),PHI(M), M=1,NUMNPT)
  480 FORMAT(4(I5,I4,1PE15.7,3X))
  500 RETURN
      END
```

DECOMP

```
C*********************************************************************
C
C  THE SUBROUTINE DECOMP OPERATES A CHOLESKI DECOMPOSITION ON A
C  VARIABLE BANDWIDTH STORAGE MATRIX (FROM JENNINGS, 1977)
C
      SUBROUTINE DECOMP(A,KDIAG)
      COMMON /AREA2/ MAXNOD,MAXELT,NUMNPT,NUMELT,NC,NF,NA,MAXA,
     *MAXCON,MAXFRE,IPRT,JAP
      INTEGER KDIAG(MAXFRE)
      DIMENSION A(MAXA)
C
      A(1)=SQRT(A(1))
      DO 1 I=2,NF
      KI=KDIAG(I)-I
      L=KDIAG(I-1)-KI+1
      DO 2 J=L,I
      X=A(KI+J)
      KJ=KDIAG(J)-J
      IF(J .EQ. 1)GO TO 2
      LBAR=KDIAG(J-1)-KJ+1
      LBAR=MAXO(L,LBAR)
      IF(LBAR .EQ. J)GO TO 2
      JJ=J-1
      DO 3 K=LBAR,JJ
    3 X=X-A(KI+K)*A(KJ+K)
    2 A(KI+J)=X/A(KJ+J)
      IF(X .LE. 0.0) THEN
        WRITE(31,20) I,X
   20   FORMAT(////,' ***** WARNING *****',//,' IN SUBROUTINE DECOMP, AT
     * I=',I5,',  X WAS :',1PE15.3,//,' THE SIGN OF X HAS BEEN CHANGED TO
     * ALLOW SQRT(X) TO BE COMPUTED.')
        X=-X
      END IF
    1 A(KI+I)=SQRT(X)
C
      RETURN
      END
```

FORW

```
C
C*************************************************************************
C
C THE SUBROUTINE FORW EXECUTES A FORWARD SUBSTITUTION IN WHICH THE
C VECTOR B IS OVERWRITTEN BY THE SOLUTION Y OF THE MATRIX EQUATION
C LY=B  (FROM JENNINGS, 1977).
C
      SUBROUTINE FORW(A,KDIAG,B)
      COMMON /AREA2/ MAXNOD,MAXELT,NUMNPT,NUMELT,NC,NF,NA,MAXA,
     *MAXCON,MAXFRE,IPRT,JAP
      INTEGER KDIAG(MAXFRE)
      DIMENSION A(MAXA),B(MAXNOD)
C
      B(1)=B(1)/A(1)
      DO 4 I=2,NF
      KI=KDIAG(I)-I
      L=KDIAG(I-1)-KI+1
      X=B(I)
      IF(L .EQ. I)GO TO 4
      II=I-1
      DO 5 J=L,II
    5 X=X-A(KI+J)*B(J)
    4 B(I)=X/A(KI+I)
C
      RETURN
      END
```

BACK

```
C
C*************************************************************************
C
C  THE SUBROUTINE BACK EXECUTES A BACKSUBSTITUTION IN WHICH THE VECTOR
C  Y, STORED IN B, IS OVERWRITTEN BY THE SOLUTION X OF THE MATRIX
C  EQUATION L(TRANSP)*X=Y (FROM JENNINGS, 1977).
C
      SUBROUTINE BACK(A,KDIAG,B)
      COMMON /AREA2/ MAXNOD,MAXELT,NUMNPT,NUMELT,NC,NF,NA,MAXA,
     *MAXCON,MAXFRE,IPPT,JAP
      INTEGER KDIAG(MAXFRE)
      DIMENSION A(MAXA),B(MAXNOD)
C
      DO 6 IT=2,NF
      I=NF+2-IT
      KI=KDIAG(I)-I
      X=B(I)/A(KI+I)
      B(I)=X
      L=KDIAG(I-1)-KI+1
      IF(L .EQ. I)GO TO 6
      II=I-1
      DO 7 K=L,II
    7 B(K)=B(K)-X*A(KI+K)
    6 CONTINUE
      B(1)=B(1)/A(1)
C
      RETURN
      END
```

```
C
C*****************************************************************
C
C     THE SUBROUTINE FLOCAL CALCULATES THE FLOW IN EACH ELEMENT,
C   THE TOTAL FLOW THROUGH THE MODEL AND THE DIRECTIONAL
C   PARAMETERS
C
      SUBROUTINE FLOCAL(NOD1,NOD2,W,ELLEN,E,PHI,IB,ELOR)
      COMMON /AREA2/ MAXNOD,MAXELT,NUMNPT,NUMELT,NC,NF,NA,MAXA,
     *MAXCON,MAXFRE,IPRT,JAP
      COMMON /AREA4/ XB(8),YB(8),HDB(8),HDE(8),ISHAP(8),IEC(8),IGEOM,N30
      COMMON /AREA8/ VISC,SPGR
      COMMON /AREA9/ DVEL(36),DFR(36),DLE(36),DLVEL(36),DLVEL2(36),
     +DPOR(18),DK(18),DLE180(18),NSEG(36)
      INTEGER IB(NUMNPT)
      INTEGER NOD1(NUMELT),NOD2(NUMELT)
      DIMENSION M2(2),DINOUT(8)
      DIMENSION W(NUMELT),ELLEN(NUMELT),E(NUMELT),ELOR(NUMELT)
      DIMENSION PHI(NUMNPT)
C
C CLEAR ARRAYS
      DO 15 I=1,NBC
   15 DINOUT(I)=0.C
      DO 20 I=1,36
        DVEL(I)=0.C
        DFR(I)=0.C
        DLE(I)=0.0
        DLVEL(I)=0.0
        DLVEL2(I)=C.0
        NSEG(I)=0
   20 CONTINUE
      DO 24 I=1,18
      DPOR(I)=0.0
      DK(I)=0.0
      DLE180(I)=0.C
24    CONTINUE
C
C PRINT HEADINGS ON DETAILED OUTPUT FILES
      IF(IPRT .LT. 2)GO TO 48
      WRITE(33,45) JAP
   45 FORMAT(////,10X,'FLOW IN ELEMENTS,     JAP=',I4,
     *///,' NUMBER',3X,'WIDTH(M)',6X,'LENGTH(M)',4X,'VELOC (M/SEC)',
     +' FL.RATE(M**3/SEC) REYNOLD NO.    DIRECTION    TRANSIT TIME',//)
   48 WRITE(31,50)
50    FORMAT('1',//,10X,'SELECTED RESULTS OF FLOW CALCULATIONS',10X,
     *' FOR SEGMENTS LOCATED AT A BOUNDARY',///,' BDRY.NO   ELEM.NO
     *      WIDTH(M)            LENGTH(M)       VELOCITY(M/SEC)    FLOW RA
     *TE(M**3/SEC)',//)
      DO 100 N=1,NUMELT
C
C CALCULATE FLOW RATE,VELOCITY AND REYNOLD NUMBER IN EACH ELEMENT
      M2(1)=NOD1(N)
      M2(2)=NOD2(N)
      THN=ELOR(N)
      WN=W(N)
```

```
  FLOCAL


        ELLENN=ELLEN(N)
        FR=E(N)*(PHI(M2(1))-PHI(M2(2)))
        VEL=FR/WN
        RE=ABS(FR/VISC)
        TRTIM=ABS(ELLENN/VEL)
        IF(IPRT .LT. 2)GO TO 55
        WRITE(33,53)N,WN,ELLENN,VEL,FR,RE,THN,TRTIM
   53 FORMAT(I5,1P7E15.7)
   55 DO 60 I=1,2
          M3=I3(M2(I))
          IF(M3 .EQ. 0)GO TO 60
          M1=I3C(M3)
          IF(M1 .EQ. 0)GO TO 60
          FR1=FR
          IF(I .EQ. 2) FR1=-FR
          WRITE(31,58) M3,N,WN,ELLENN,VEL,FR1
   58     FORMAT(2(I5,5X),1P4E20.7)
          DINOUT(M3)=DINOUT(M3) + FR1
   60 CONTINUE
C
C  COMPUTE DIRECTIONAL POROS. & PERM. (0-180 DEGREES)
        IDIR=INT(THN/10.) + 1
        DPOR(IDIR)=DPOR(IDIR) + WN*ELLENN
        DK(IDIR)=DK(IDIR) + WN*WN*ELLENN
        DLE180(IDIR)=DLE180(IDIR) + ELLENN
C  COMPUTE OTHER DIRECTIONAL PARAMETERS (0-360 DEGREES)
        IF(FR .LE. 0.0) THN=THN+180.
        IDIR=INT(THN/10.)+1
        AVEL=ABS(VEL)
        AFR=ABS(FR)
        DVEL(IDIR)=DVEL(IDIR)+AVEL
        DFR(IDIR)=DFR(IDIR)+AFR
        DLE(IDIR)=DLE(IDIR)+ELLENN
        DLVEL(IDIR)=DLVEL(IDIR)+ELLENN*AVEL
        DLVEL2(IDIR)=DLVEL2(IDIR) + ELLENN*AVEL*AVEL
        NSEG(IDIR)=NSEG(IDIR)+1
  100 CONTINUE
C
C  PRINT SUMMARY OUTPUT FILES
        CALL OUTPT4(DINOUT)
        RETURN
        END
```

```
C
C*********************************************************************
C
      SUBROUTINE OUTPT4(DINOUT)
C THIS SUBROUTINE PRINTS OUT A SUMMARY OF FLOW CALCULATIONS
C AND VALUES OF DIRECTIONAL PARAMETERS
C
      COMMON /AREA2/ MAXNOD,MAXELT,NUMNPT,NUMELT,NC,NF,NA,MAXA,
     *MAXCON,MAXFRE,IPRT,JAP
      COMMON /AREA4/ XB(8),YB(8),HDB(8),HDE(8),ISHAP(8),IBC(8),IGEOM,NBO
      COMMON /AREA9/ DVEL(36),DFR(36),DLE(36),DLVEL(36),DLVEL2(36),
     +DPOR(18),DK(18),DLE180(18),NSEG(36)
      DIMENSION DINOUT(8)
      CHARACTER BCD*26
C
C PRINT SUMMARY OF FLOW CALCULATIONS
      WRITE(31,110) JAP
  110 FORMAT('1',20X,'SUMMARY OF FLOW CALCULATIONS FOR JAP=',I4,
     *////,' TOTAL FLOW ALONG THE BOUNDARIES',///,
     *' BDRY NO',5X,'FLOW (M**3/SEC) (- =>FLOW OUT)'//)
      WRITE(31,115) (I,DINOUT(I), I=1,NBO)
  115 FORMAT(I5,1PE20.10)
      IF(IGEOM .EQ. 1) GO TO 125
      DELTAFR=DINOUT(3)+DINOUT(6)
      AVGFR=(DINOUT(3)-DINOUT(6))/2.0
      EORFR=DELTAFR/AVGFR
      WRITE(31,117) DELTAFR,AVGFR,EORFR
  117 FORMAT(//,2X,'DIFFERENCE IN FLOW RATE BETWEEN UP AND DOWNSTREAM',
     *//,14X,'BOUNDARIES  3  AND  7 ',//,10X,'DELTA(FL.RA)',11X,
     *'AVG(FL.RA)',7X,'RELATIVE ERROR(FL.RA)',//,5X,
     *3(8X,1P1E12.3))
C
C PRINT DIRECTIONAL PARAMETERS
125   WRITE(31,126)
126   FORMAT(///,10X,'DIRECTIONAL POROSITY AND PERMEABILITY',///,
     *' DIRECTION          LENGTH        POROSITY     PERMEABILITY',//,
     *' (DEGR/10)          (M)             (M**2)         (M**3)',///)
      WRITE(31,128) (I,DLE180(I),DPOR(I),DK(I),I=1,18)
128   FORMAT(I5,5X,1P3E15.2)
      WRITE(31,130)
130   FORMAT('1',10X,'OTHER DIRECTIONAL PARAMETERS',///,
     *' DIRECTION        NSEG  (1)VELOC  (2)FL.RA  (3)LENGTH',
     *'(3)X(1) (3)X(1)**2',//,
     *' (DEGR/10)    ',10X,'(M/S)     (M**3/S)',7X,'(M)',///)
      WRITE(31,135) (I,NSEG(I),DVEL(I),DFR(I),DLE(I),
     *DLVEL(I),DLVEL2(I),I=1,36)
  135 FORMAT(I5,5X,I10,1P5E10.3)
      WRITE(32) DVEL,DLVEL,DLVEL2,DFR,DLE,NSEG
C
C CONSTRUCT ROSE DIAGRAMS
      IF(IPRT-2) 150,140,138
138   BCD='VELOCITIES (M/S)'
      CALL ROSE(DVEL,BCD)
      BCD='SEGMENT LENGTHS (M)'
      CALL ROSE(DLE,BCD)
```

```
OUTPT4

      BCD='LENGTH X VELOC**2'
      CALL ROSE(DLVEL2,BCD)
140   BCD='FLOW RATES (M**3/S)'
      CALL ROSE(DFR,BCD)
  150 CONTINUE
      RETURN
      END
```

```
C*****************************************************************************
C                                                                             *
C       THIS SUBROUTINE PRINTS A ROSE DIAGRAM FOR A DIRECTIONAL               *
C     PARAMETER                                                               *
C                                                                             *
C*****************************************************************************
      SUBROUTINE ROSE(R2,BCD1)
      DIMENSION R(36),R2(36),COSTH(36),SINTH(36)
      CHARACTER LPLOT(80,64)*1,BCD1*26,N1*1,N2*1,N3*1
      COMMON /AREA12/ LPLOT
      N1='.'
      N2='0'
      N3='*'
      COUNT=0.0
      IBRANCH=1
      DO 10 J=1,64
        DO 5 I=1,79,2
          LPLOT(I,J)=' '
          LPLOT(I+1,J)=' '
    5   CONTINUE
   10 CONTINUE
C
C COMPUTE THE ARRAYS COSTH() AND SINTH()
      DO 20 I=1,36
        IDEG=I*10
        FDEG=FLOAT(IDEG)-0.5
        COSTH(I)=COSD(FDEG)
        SINTH(I)=SIND(FDEG)
   20 CONTINUE
C
      WRITE(31,25) BCD1
   25 FORMAT('1',//,'T',//,20X,'ROSE OF ',A,//)
      CALL MXMNAS(R2,36,RMAX,RMIN,AVG,SIGMA,25)
C
C    DRAW A CIRCLE WITH A MARK EVERY 10 DEGREES
      DO 30 I=1,36
   30 CALL MARK(COSTH(I)*.95,SINTH(I)*.95,N1)
C
C    SCALE ARRAYS
      DO 100 I=1,36
        R(I)=R2(I)/(RMAX*1.05)
        IF(R(I).LT..01) R(I)=.01
        IF(R(I).GT.0.1) COUNT=COUNT+1.0
  100 CONTINUE
C
C    FOR COUNT LESS THAN 3 USE LOG VALUES
      IF(COUNT.LT.3.) IBRANCH=2
      DO 300 I=1,36
        IF(R(I).LT.0.02) GOTO 300
        IF(I.EQ.1) THEN
          C1=1.0
          S1=0.0
        ELSE
          C1=COSTH(I-1)
          S1=SINTH(I-1)
```

```
        ENDIF
        GOTO(150,160) IBRANCH
  150   X1=R(I)*C1
        Y1=R(I)*S1
        X2=R(I)*COSTH(I)
        Y2=R(I)*SINTH(I)
        GOTO 200
  160   X1=(C1*LOG10(R(I)*100))/2.0
        Y1=(S1*LOG10(R(I)*100))/2.0
        X2=(COSTH(I)*LOG10(R(I)*100))/2.0
        Y2=(SINTH(I)*LOG10(R(I)*100))/2.0
C
C    DRAW A LINE BETWEEN THE TWO POINTS
  200   CALL MARK(X1,Y1,N3)
        CALL MARK(X2,Y2,N3)
        DX=(X1-X2)/5.0
        DY=(Y1-Y2)/5.0
        RINC=0.0
  225   RINC=RINC+1
        XM=X2+DX*RINC
        YM=Y2+DY*RINC
        CALL MARK(XM,YM,N3)
        IF(RINC.LT.4.0) GOTO 225
  300 CONTINUE
      CALL MARK(0.,0.,N2)
      IF(IBRANCH.EQ.2) THEN
        WRITE(31,40)
   40   FORMAT(20X,'ROSE OF LOG VALUES',//)
      ENDIF
      DO 400 J=64,1,-1
        WRITE(31,80) (LPLOT(I,J),I=1,80)
   80   FORMAT(1X,80A1)
  400 CONTINUE
      RETURN
      END
```

MXMNAS

```
C*************************************************************************
C                                                                        *
C       THIS SUBROUTINE CALCULATES THE MAXIMUM,MINIMUM,AVERAGE AND       *
C     STANDARD DEVIATION FOR A SET OF DATA                               *
C                                                                        *
C*************************************************************************
      SUBROUTINE MXMNAS(ARRAY,NUM,XMAX,XMIN,TAVG,SIGMA,L)
      DIMENSION ARRAY(NUM)
C
C  INITIALIZE VARIABLES
      TTOTAL=ARRAY(1)
      T2TOTAL=ARRAY(1)*ARRAY(1)
      XMAX=ARRAY(1)
      XMIN=ARRAY(1)
      RNUM=FLOAT(NUM)
      DO 100 I=2,NUM
        IF(ARRAY(I).LT.XMIN) XMIN=ARRAY(I)
        IF(ARRAY(I).GT.XMAX) XMAX=ARRAY(I)
        TTOTAL=TTOTAL+ARRAY(I)
        T2TOTAL=T2TOTAL+ARRAY(I)*ARRAY(I)
  100 CONTINUE
      TAVG=TTOTAL/RNUM
      T2AVG=T2TOTAL/RNUM
      SIGMA=SQRT(T2AVG-(TAVG*TAVG))
      IF(L.GT.0) THEN
      WRITE(L,6)
    6 FORMAT(23X,'BASIC DESCRIPTIVE STATISTICS',//,5X,'NUMBER OF',
     *15X,'STANDARD',//,4X,'DATA POINTS    MEAN       DEVIATION',
     *4X,'MAXIMUM      MINIMUM',//)
      WRITE(L,7) NUM,TAVG,SIGMA,XMAX,XMIN
    7 FORMAT(7X,I4,4X,4(1PE11.3,1X),///)
      ENDIF
      RETURN
      END
```

MARK

```
C
C*****************************************************************
C
C     THE SUBROUTINE MARK INSERTS A MARK INTO THE LOCATION (X,Y).
C   THE MARK USED IS THAT DEFINED BY THE CHARACTER NM. NOTE THAT
C   BOTH X AND Y MUST BOTH BE IN THE RANGE OF -1 TO +1.
C
      SUBROUTINE MARK(X,Y,NM)
      COMMON/AREA12/ LPLOT
      CHARACTER NM*1,LPLOT(80,54)*1
C
C   SCALE X AND Y
      IX=INT((X*40)+40.5)
      IY=INT((Y*32)+32.5)
      IF(IX.LT.1) IX=1
      IF(IY.LT.1) IX=1
      LPLOT(IX,IY)=NM
      RETURN
      END
```

G.4 Liste du programme NETRANS

NETRANS

```
      PROGRAM NETRANS
C*****************************************************************************
C
C  PROGRAM TO SIMULATE STOCHASTICALLY THE MIGRATION OF PARTICLES
C  THROUGH A FRACTURE NETWORK BASED ON STATISTICS OF THE DIRECTIONAL
C  PARAMETERS
C
C WRITTEN BY A. ROULEAU AND D. LENTZ, 1984; UPDATE BY A.ROULEAU, JANUARY 1986
C
C*****************************************************************************
      COMMON /AREA1/ COSTH(36),SINTH(36),DLM(36),DMTVM(36),RDFR(36),
     +XLEN,YLEN,TRAVDIS,IDIR(100),MPART(10),IBC1,IPRT,JAP
      COMMON/AREA2/TITLE
      DIMENSION DVEL(36),DFR(36),DLE(36),DLVEL(36),DLVEL2(36),NSEG(36)
      CHARACTER TITLE(20)*4,FMT(5)*20
      DATA (FMT(I),I=1,5)/'(2(/),10I3)','(4(/),10I3)','(6(/),10I3)',
     *'(8(/),10I3)','(10(/),10I3)'/
      MAXNPRT=50
C
C  READ IN INPUT DATA
      READ(1,5)TITLE
    5 FORMAT(20A4)
      WRITE(41,6) TITLE
6     FORMAT('1',//,1X,20A4,//,' FILE : TRANSIT.U41')
      READ(1,10) NSET,IGEOM,NAP,NAAP,IPRT,NPART,TRAVDIS,NMAP
   10 FORMAT(I5,20X,3I5,10X,I5,I5,F5.0,I3)
      NAPER=NAP*NAAP
      IF(IGEOM .EQ. 2) THEN
        WRITE(41,12)
12      FORMAT(///,' ****  SORRY ! *****',//,10X,
     +  'THIS VERSION OF NETRANS DOES NOT APPLY TO CIRCULAR GEOMETRY')
        STOP
      ENDIF
      IF(NPART.GT.MAXNPRT) THEN
        WRITE(41,15) NPART,MAXNPRT
   15   FORMAT('1',10X,'***ERROR AT INPUT STAGE***',//,10X,
     +  'NPART=',I5,' ,  THIS IS LARGER THAN MAXNPRT=',I5)
        STOP
      ENDIF
      READ(1,20) XMINI,YMAXI,XMAXI,YMINI
   20 FORMAT(20X,4F5.0)
      XLEN=XMAXI-XMINI
      YLEN=YMAXI-YMINI
      READ(1,40) IBC1
   40 FORMAT(20X,I5,7(/))
      IF(NMAP.GT.0) THEN
        READ(1,FMT(NSET))(MPART(I),I=1,NMAP)
      ENDIF
        DO 70 I=1,36
          IDEG=I*10
          FDEG=FLOAT(IDEG)-5.0
          COSTH(I)=COSD(FDEG)
          SINTH(I)=SIND(FDEG)
   70   CONTINUE
C
```

```
C   READ IN DIRECTIONAL PARAMETERS
       DO 400 JAP=1,NAPER
         READ(32) DVEL,DLVEL,DLVEL2,DFR,DLE,NSEG
C
C   COMPUTE AUXILIARY STATISTICS OF DIRECTIONAL PARAMETERS
         SUMFR=0.0
         DO 50 I=1,36
    50   SUMFR=SUMFR+DFR(I)
         IF(SUMFR.EQ.0.0)THEN
           WRITE(41,56)
56         FORMAT(////,1X,'SUMFR=0.0 : ERROR PR BABLY IN INPUT ',
      +    'DIRECTIONAL PARAMETERS')
           STOP
         ENDIF
         DO 60 I=1,36
           IF(NSEG(I).EQ.0) THEN
             RDFR(I)=0.0
             DLM(I)=0.0
           ELSE
             RDFR(I)=DFR(I)/SUMFR
             DLM(I)=DLE(I)/NSEG(I)
             DWTVM(I)=DLVEL2(I) / DLVEL(I)
           ENDIF
    60   CONTINUE
C
C COMPUTE IDIR(J), THE DIRECTION (1 TO 36) A PARTICLE SHALL
C TAKE FOR CORRESPONDING VALUE OF RANDOM J (1 TO 100)
         DO 71 I=1,100
71       IDIR(I)=0
         IR=1
         DO 100 I=1,36
           IF (RDFR(I).LE.0.005) GOTO 100
           R100=RDFR(I)*100.0
           IR100=NINT(R100)
           IREND=IR+IR100-1
           IF(IREND.GT.100) IREND=100
           DO 90 J=IR,IREND
    90     IDIR(J)=I
           IR=IREND+1
   100   CONTINUE
C
C PRINT OUT PARTICLE TRANSIT PARAMETERS
         CALL OUTPT5
C
C   COMPUTE TRANSIT TIME OF PARTICLES THROUGH THE NETWORK
         CALL TTIMER(NPART,NMAP)
   400 CONTINUE
       STOP
       END
```

```
C
C ******************************************************************
C
      SUBROUTINE OUTPT5
C
C THIS SUBROUTINE PRINTS OUT INFORMATIONS ON PARTICLE TRANSIT
C PARAMETERS.
C
      COMMON /AREA1/ COSTH(36),SINTH(36),DLM(36),DWTVM(36),RDFR(36),
     +XLEN,YLEN,TRAVDIS,IDIR(100),MPART(10),IBC1,IPRT,JAP
      CHARACTER BCD1*4
C
      WRITE(41,10) JAP
10    FORMAT(///,10X,'AUXILIARY DIRECTIONAL PARAMETERS FOR JAP=',I4)
      WRITE(41,20)
  20  FORMAT(/,1X,'DIRECTION',5X,'RDFR',7X,'DLM',7X,'DWTVM',//)
      WRITE(41,30) (I,RDFR(I),DLM(I),DWTVM(I),I=1,36)
  30  FORMAT(2X,I5,5X,1P3E10.2)
C
      WRITE(41,40)
40    FORMAT(////,10X,'ARRAY IDIR',//,10X,'FORMAT: 10(J  IDIR(J))',///)
      WRITE(41,50) (J,IDIR(J), J=1,100)
50    FORMAT(10(3X,2(I4)))
C
      IF(IBC1 .EQ. 1) THEN
        BCD1=' X '
      ELSE
        BCD1=' Y '
      ENDIF
      WRITE(41,60) XLEN,YLEN,TRAVDIS,BCD1
60    FORMAT(///,10X,'SIZE OF GENERATED NETWORK :',//,15X,'X-LENGTH=',
     +F10.4,' (M)',//,15X,'Y-LENGTH=',F10.4,' (M)',///,5X,
     +'PARTICLES MUST TRAVEL ',F6.2,' (M) IN THE ',A4,' DIRECTION')
C
      RETURN
      END
```

```
C*****************************************************************************
C
C      SUBROUTINE TTIMER COMPUTES STOCHASTICALLY THE TRANSIT TIME OF
C   PARTICLES THROUGH THE NETWORK USING THE STATISTICS OF THE
C   DIRECTIONAL PARAMETERS (RECTANGULAR MODEL)
C
C*****************************************************************************
      SUBROUTINE TTIMER(NPART,NM)
      COMMON /AREA1/ COSTH(36),SINTH(36),DLM(36),DWTVM(36),RDFR(36),
     +XLEN,YLEN,TRAVDIS,IDIR(100),MPART(10),ISC1,IPRT,JAP
      COMMON/AREA2/TITLE
      DIMENSION R(100),XTRAV(50),YTRAV(50),TIM(50)
     *,NSTEP(50),TRAV(50)
      DOUBLE PRECISION DSEED
      CHARACTER TITLE(20)*4,BCD1*22
C
      NR=100
      DSEED=1.DO
      ICOUNT=1
      NTEST=0
      NMAP=NM
      IF(NMAP.GT.0) THEN
        WRITE(41,20)
20      FORMAT(///,5X,'TRACKING OF SELECTED PARTICLES',///,6X,
     *  ' STEP',6X,'CUMX',6X,'CUMY',6X,'CUMT',//)
      ENDIF
C
C  COMPUTE TRAVEL TIME FOR NPART PARTICLES
      DO 150 NT=1,NPART
        CUMX=0.0
        CUMY=0.0
        CUMT=0.0
        CUML=0.0
        NSTEP(NT)=0
        IF(NMAP.GT.0) THEN
          IF(NT.EQ.MPART(ICOUNT)) THEN
            ICOUNT=ICOUNT+1
            IF(ICOUNT.GT.NMAP) THEN
              NMAP=0
            ENDIF
          NTEST=NT
          WRITE(41,35) NT
35        FORMAT(//,13X,'PARTICLE NUMBER ',I4,//,)
          ENDIF
        ENDIF
25      CALL GGUBS(DSEED,NR,R)
        DO 90 I=1,100
C
C  DETERMINE DIRECTION CORRESPONDING TO THE RANDOM NUMBER R(I)
          R100=R(I)*100.0
          IR100=INT(R100) +1
          JDIR=IDIR(IR100)
C
C  COMPUTE TRAVELLED DISTANCE AND TRAVEL TIME
          DL=DLM(JDIR)
```

```
            DX=DL*COSTH(JDIR)
            DY=DL*SINTH(JDIR)
            DT=DL/DWTVM(JDIR)
            CUMX=CUMX+DX
            CUMY=CUMY+DY
            CUMT=CUMT+DT
            CUML=CUML+DL
            IF(NT.EQ.NTEST) THEN
              NSTP=I+NSTEP(NT)*100
              WRITE(41,45) NSTP,CUMX,CUMY,CUMT
   45         FORMAT(7X,I3,3X,1P3E10.2)
            ENDIF
C
C CHECK IF TRAVELLED DISTANCE GREATER THAN TRAVDIS
            IF(IBC1.EQ.1)THEN
              IF(ABS(CUMX).GE.TRAVDIS)THEN
                GOTO 120
              ENDIF
            ELSE
              IF(ABS(CUMY).GE.TRAVDIS)THEN
                GOTO 120
              ENDIF
            ENDIF
   90     CONTINUE
            NSTEP(NT)=NSTEP(NT)+1
          GOTO 25
C
C TRIM THE TRAVELLED PATH AT LIMIT OF SYSTEM
120       IF(IBC1 .EQ. 1) THEN
            XOUT=ABS(CUMX)-TRAVDIS
            XOUT=SIGN(XOUT,CUMX)
            DOUT=XOUT/COSTH(JDIR)
            YOUT=DOUT*SINTH(JDIR)
          ELSE
            YOUT=ABS(CUMY)-TRAVDIS
            YOUT=SIGN(YOUT,CUMY)
            DOUT=YOUT/SINTH(JDIR)
            XOUT=DOUT*COSTH(JDIR)
          ENDIF
          XTRAV(NT)=CUMX-XOUT
          YTRAV(NT)=CUMY-YOUT
          TRAV(NT)=CUML-DOUT
          TIM(NT)=CUMT-DOUT/DWTVM(JDIR)
          NSTEP(NT)=NSTEP(NT)*100+I
  150   CONTINUE
C
C PRINT STATISTICS OF ALL PARTICLES
        WRITE(41,201) JAP
201     FORMAT('1',///,10X,'STATISTICS OF ALL PARTICLES FOR JAP=',I4,//
       *,5X,'NT',10X,'XTRAV(M)  YTRAV(M)   TRAV(M)   TIME(S)',5X,
       +'NSTEP',/)
        WRITE(41,202)(I,XTRAV(I),YTRAV(I),TRAV(I),TIM(I),NSTEP(I),
       *I=1,NPART)
  202   FORMAT(2X,I5,3X,1P4E10.2,2X,I6)
C
C COMPUTE BASIC STATISTICS AND MAKE FREQUENCY DIAGRAMS OF RESULTS
        HCD1='ABS.TRAV.DIST.,TRAV(M)'
```

```
TTIMER

      CALL MXMNAS(TRAV,NPART,XMAX,XMIN,TAVG,SIGMA,41,BCD1)
      BCD1='TRANSIT TIME (S)'
      CALL HSTGR(TIM,NPART,BCD1)
      RETURN
      END
```

```
C
C*****************************************************************************
C
      SUBROUTINE HSTGR(ARRAY,NUM,T1)
C
C THIS SUBROUTINE PLOTS HORIZONTAL-BAR FREQUENCY HISTOGRAMS OF THE
C THE ARRAY ARRAY.
C
      DIMENSION ARRAY(NUM),FREQ(21)
      CHARACTER T1*22,A(101)*1
C
C CALCULATE BASIC STATISTICS AND PRINT HEADINGS
      WRITE(*,5) T1
5     FORMAT('1',15X,'FREQUENCY DIAGRAMS OF  ',A,//)
      CALL XMMAS(ARRAY,NUM,XMAX,XMIN,TAVG,SIGMA,41,T1)
      WRITE(41,10)
   10 FORMAT(//,1X,'MIDDLE    FREQ  CUM REL',13X,'RELATIVE',
     *' FREQUENCY DIAGRAM',24X,'CUMULATIVE FREQUENCY',
     +' DIAGRAM',//,1X,'INTERV',9X,'FREQ',//)
C
      RANGE=XMAX-XMIN
      XINT=RANGE/20.0
      YC=0.0
      I=1
      C1=50.0/FLOAT(NUM)
      DO 50 I=1,21
   50 FREQ(I)=0.0
C
C DISPATCH ENTRIES INTO APPROPRIATE RANGE
      DO 150 I=1,NUM
        VAR=ARRAY(I)
        N=INT((VAR-XMIN)/XINT)+1
        FREQ(N)=FREQ(N)+1.0
  150 CONTINUE
      DO 300 I=1,21
        X=XMIN+XINT*(FLOAT(I)-.5)
        LY=NINT(FREQ(I))
        YC=YC+FREQ(I)*C1
        LYC=NINT(YC)
C
C  CREATE PRINTING ARRAY
        DO 250 IT=1,101
          ITEST=NINT(FREQ(I)*C1)
          IF(IT.GT.51)ITEST=(LYC+51)
          IF(IT.LE.ITEST) THEN
            A(IT)='*'
          ELSE
            A(IT)=' '
          ENDIF
  250   CONTINUE
C
C  PRINT DIAGRAMS
        A(51)='1'
        WRITE(*,225) X,LY,(LYC+LYC),(A(J),J=1,101)
  225   FORMAT(1X,1PE8.2,1X,I3,1X,I5,3X,'1',101A)
```

```
*STGR

 300 CONTINUE
       RETURN
       END
```

MXMNAS

```
C**********************************************************************
C                                                                     *
C        THIS SUBROUTINE CALCULATES THE MAXIMUM,MINIMUM,AVERAGE AND   *
C      STANDARD DEVIATION FOR A SET OF DATA                           *
C                                                                     *
C**********************************************************************
      SUBROUTINE MXMNAS(ARRAY,NUM,XMAX,XMIN,TAVG,SIGMA,L,T1)
      DIMENSION ARRAY(NUM)
      CHARACTER T1*22
C
C  INITIALIZE VARIABLES
      TTOTAL=ARRAY(1)
      T2TOTAL=ARRAY(1)*ARRAY(1)
      XMAX=ARRAY(1)
      XMIN=ARRAY(1)
      RNUM=FLOAT(NUM)
      DO 100 I=2,NUM
        IF(ARRAY(I).LT.XMIN) XMIN=ARRAY(I)
        IF(ARRAY(I).GT.XMAX) XMAX=ARRAY(I)
        TTOTAL=TTOTAL+ARRAY(I)
        T2TOTAL=T2TOTAL+ARRAY(I)*ARRAY(I)
  100 CONTINUE
      TAVG=TTOTAL/RNUM
      T2AVG=T2TOTAL/RNUM
      SIGMA=SQRT(T2AVG-(TAVG*TAVG))
      IF(L.GT.0) THEN
      WRITE(L,5) T1
5     FORMAT(/,1X,'BASIC DESCRIPTIVE STATISTICS OF ',A,/,
     +5X,'NUMBER OF',15X,'STANDARD',/,4X,'DATA POINTS     MEAN',
     +6X,'DEVIATION',4X,'MAXIMUM      MINIMUM',/)
      WRITE(L,7) NUM,TAVG,SIGMA,XMAX,XMIN
    7 FORMAT(7X,I4,4X,4(1PE11.3,1X))
      ENDIF
      RETURN
      END
```